Description and Simulation of Visual Texture

Carstensen, Jens Michael

Publication date:
1992

Document Version
Publisher's PDF, also known as Version of record

DESCRIPTION AND SIMULATION OF VISUAL TEXTURE

By
Jens Michael Carstensen

LYNGBY 1992
Ph.D. THESIS
NO. 59

ISSN 0107–525X

Printed by IMSOR, Technical University of Denmark

This document was formatted with LaTeX.

HIPS and HIPS-2 are trademarks of SharpImage Software, New York, and The Turing Institute, Glasgow, UK.

CART is a trademark of California Statistical Software, Inc., Lafayette, California.

SAS is a registered trademark of SAS Institute Inc., Cary, North Carolina.

Connection Machine and *C** are registered trademarks of Thinking Machines Corporation. CM-200 is a trademark of Thinking Machines Corporation.

HP-UX and HP Apollo 9000/750 are registered trademarks of Hewlett-Packard Company.

Sun-4 is a registered trademark of Sun Microsystems, Inc.

Some of the work in this thesis has previously been published in:

Carstensen, J.M. & Conradsen, K. (1992) Spin-flip alternatives to spin-exchange Markov random field simulation implemented on a SIMD massively parallel computer, Submitted to IEEE Transactions on Pattern Analysis and Machine Intelligence.

Carstensen, J.M., Grunkin, M. & Conradsen, K. (1991) Measurement of enzymatic treatment effect on textile using digital image analysis, IMSOR Technical report.

Preface

This thesis has been prepared at the Institute of Mathematical Statistics and Operations Research (IMSOR), Technical University of Denmark, in partial fulfillment of the requirements for the degree of Ph.D. in engineering.

The general framework of the thesis is statistics and digital image analysis. It is implied that the reader has a basic knowledge of these areas.

The treatment of the subjects is by no means exhaustive, but is intended to improve the knowledge on texture description and texture simulation by going through selected theory and examples. Hopefully this can lead to an improved texture understanding.

Lyngby, April 1992

Jens Michael Carstensen

Acknowledgements

The author would like to thank Professor Knut Conradsen for guidance and encouragement during the course of this work and for providing excellent research facilities for the image group at IMSOR.

Professor Brian D. Ripley is thanked for inspiring discussions during my visit to the University of Oxford and for providing many useful references.

I wish to thank my colleagues at IMSOR for contributing to a pleasant and inspiring scientific and social environment. Especially I would like to thank my officemate Michael Grunkin for help and moral support and for coping with my frustrations in a pleasant and enjoyable way. Dr. Bjarne Kjær Ersbøll provided useful help and comments, for which I am thankful. Niels Christian Krieger Lassen, Niels Jacob Carstensen and Allan Aasbjerg Nielsen were very helpful in the last critical moments of the preparation of this thesis. I appreciate the help and continuous supply of LaTeX-styles from Dr. Carl M. Bilbo. Dr. Niels Kjølstad Poulsen is acknowledged for his expert comments on control theory. I am also very grateful to the members of the image group, not mentioned above, for help and inspiration.

Image data was kindly provided by Novo Nordisk and Imperial Cancer Research Fund. I wish to thank Peter Rosholm of Novo Nordisk, and Drs. Richard Mott and Hans Lehrach of Imperial Cancer Research Fund, London for their splendid collaboration.

Dr. Peter Frykman of the Geological Survey of Denmark is acknowledged for his collaboration on our attempt to make a reservoir simulation program that proves useful to geologists.

I am grateful to the Viggo Jarl foundation for supporting me financially during my studies.

This research was partially sponsored by the Danish Technical Research Council and the Danish Natural Science Research Council under the MOBS and CAP programs.

Summary

The problem of texture analysis is considered within the framework of digital image analysis. An extensive set of texture statistics is reviewed and explained, and their performance in measuring enzymatic treatment effect on textile and in classification of a more general set of textures is studied. We found that both problems were solved satisfactorily with the set of texture statistics used.

Markov random fields are reviewed and investigated as models of texture. Results from the field of statistical physics are reformulated in a statistical setting. Standard Markov random fields do not have the ability to model morphological properties of textures, and this leads us to formulate an extension in the terms of mathematical morphology. The properties of morphological Markov random fields are illustrated. We go through the problem of Markov random field parameter estimation and suggest an extension of the asymptotic maximum likelihood estimator (Pickard, 1987) to the anisotropic first-order model.

Markov random field simulation is described and a new, fast, parallel algorithm for simulation conditional on the first-order statistics is presented. This algorithm and the morphological Markov random fields are then used for the simulation of the geometrical structure of oil reservoirs.

Markov random fields in a Bayesian setting are used successfully to analyze hybridization filters automatically for the human genome project.

Resumé

Teksturanalyse betragtes inden for rammerne af digital billedanalyse. Et omfattende antal statistiske stikprøvefunktioner til teksturbeskrivelse er gennemgået og forklaret, og deres evne til at måle effekt af enzymbehandling på tekstiler og til at klassificere et mere generelt sæt af teksturer er undersøgt. Begge problemer viste sig at kunne løses tilfredsstillende med de anvendte stikprøvefunktioner.

Der gives en oversigt over Markovfelter, og deres anvendelighed som teksturmodeller udforskes. Resultater om disse modeller fra statistisk fysik reformuleres i statistisk terminologi. Sædvanlige Markovfelter kan ikke modellere teksturers morfologiske egenskaber, og dette fører os frem til at formulere en udvidelse ved brug af matematisk morfologi. Morfologiske Markovfelters egenskaber bliver endvidere illustreret. Der redegøres for problemerne i forbindelse med estimation af parametre, og der foreslås en udvidelse af den asymptotiske maximum likelihood estimator (Pickard, 1987) til det anisotrope tilfælde.

Teorien for simulation af Markovfelter er gennemgået, og der præsenteres en ny, hurtig, parallel algoritme til simulation af Markovfelter givet den marginale fordeling af pixelværdier. Denne algoritme og de morfologiske Markovfelter bliver derefter anvendt til simulation af den rumlige struktur i oliereservoirer.

Markovfelter bliver i en Bayesiansk sammenhæng anvendt til automatisering af analysen af hybridiseringsfiltre indenfor *human genome* projektet.

Contents

Chapter 1

Introduction

1.1 Texture

The term *visual texture* in the title of this thesis emphasizes, that the definition of texture used here is closely related to perception.

> *A texture is a region in 2D or 3D that can be perceived as being spatially homogeneous in some sense.*

This definition is very broad. It includes as texture the totally uniform region, which in the daily language is said to have no texture. Indeed the interesting thing about textures is the study of the spatial variations over the textured region, and these variations often become synonymous with

texture. We emphasize that textures only differing in luminance are considered different textures according to our definition. Brodatz (1966) is a photographic album with 112 textures. This album has become a standard reference in texture analysis and subsequently these textures shall be referred to as the Brodatz textures .

Figure 1.1 shows a strictly random texture. The pixels are uncorrelated. Figure 1.2 shows a strictly deterministic texture (a checkerboard). It is a strictly ordered pattern, that is fully determined from the knowledge of a small subpattern. Observable textures are somewhere between these two extremes. Figures 1.3 and 1.4 show examples of a random texture (handmade paper) and a deterministic texture (a brick wall). The word texture comes from the Latin word *textura*, that means textile fabric, and textile fabric is another example of a deterministic texture. Rao (1990) classifies all of the Brodatz textures in three classes: disordered (random), weakly ordered, and strongly ordered (deterministic).

The question of *scale* or *resolution* is fundamental to texture perception. If we zoom in on the brick wall of figure 1.4 we see the texture of the individual bricks. If we zoom out we may see a texture of wall shading or a texture of wall and windows. Thus there may be several levels of completely different textures in the same image but at different scales. A texture with more than one texture level is called a *hierarchical texture* . To distinguish between different texture levels we can use the terms *microtexture* and *macrotexture*. Subsequently the terms scale and resolution will be used interchangeably.

Researchers in texture perception have investigated preattentive (effortless or instantaneous) texture discrimination in the human visual system. The famous *iso-second-order conjecture* (Julesz, 1975) stated that textures with

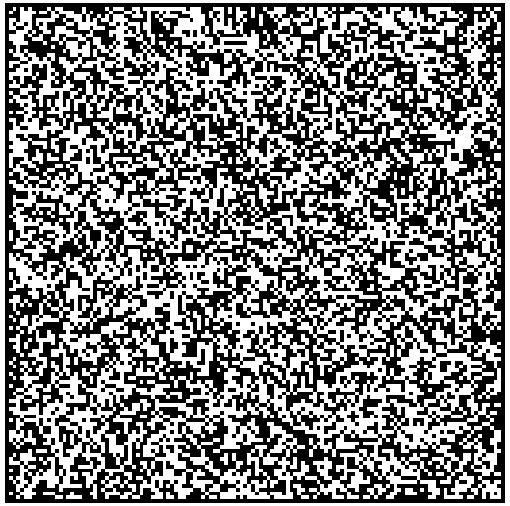

Figure 1.1. A strictly random texture

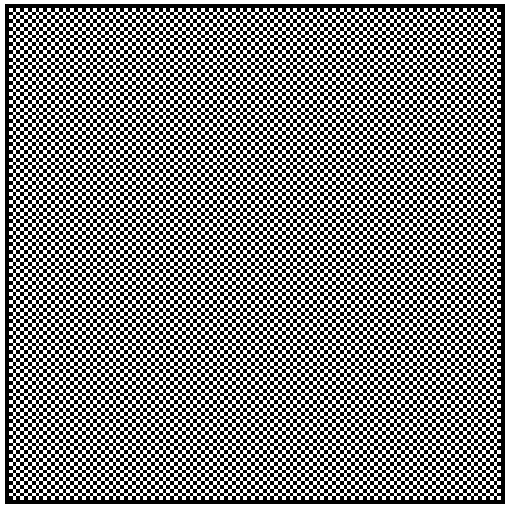

Figure 1.2. A strictly deterministic texture

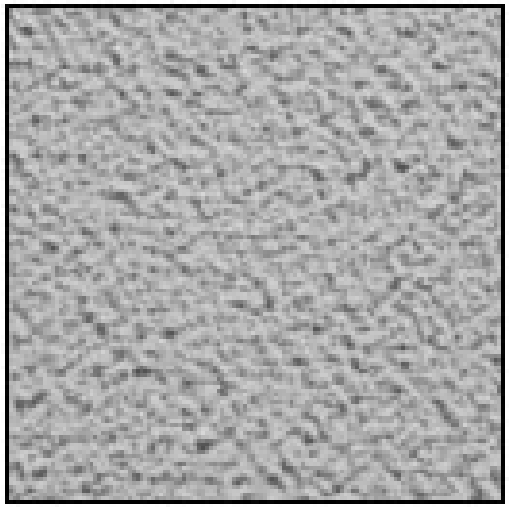

Figure 1.3. Handmade paper (D57 from Brodatz)

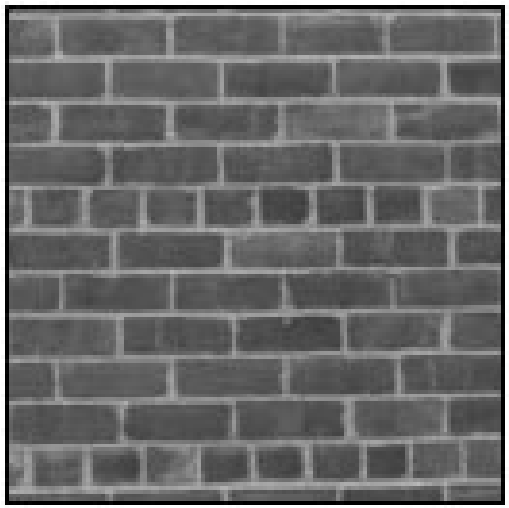

Figure 1.4. Brick wall (D95 from Brodatz)

the same second-order statistics (gray level statistics of pairs of pixels) can not be distinguished even if they have different third- or higher-order statistics. This conjecture has later been disproved (Julesz, 1981) and replaced by a *texton* theory (Julesz & Bergen, 1983). Textons are small conspicuous features like

- Elongated shapes, such as ellipses, rectangles or line segments.
- Ends of line segments.
- Crossings of line segments.

The texton conjecture argues that preattentive texture discrimination is based on differences in the density of textons.

1.2 Texture analysis

The main goal of *texture analysis* is to extract useful textural information from an image. Historically there has been two major approaches, a *structural* and a *statistical*. The structural approach describes a texture by a subpattern or *primitive* and the spatial distribution of primitives, the *placement rule*. The primitives are also called *texture elements*. If we consider the brick wall the primitive is a brick and the placement rule specifies the arrangement of bricks in the wall. The statistical approach is more generally applicable, because it does not presume that the texture can be described in terms of primitives and placement rules. It draws on the general set of statistical tools. This thesis is primarily based on the statistical approach.

The extraction of texture features is essential to applications such as *texture measurement*, *texture summarization*, *texture classification* and *texture segmentation* (*Texture description* denotes all of these areas). The goal of texture measurement is to characterize a texture with one feature, e.g. a feature for textile wear assessment. In texture summarization we give summaries reflecting the visual properties of textures. Texture classification usually serves one of two goals. We may want to assign a class to an entire texture, e.g. accept or reject in industrial quality control. We may also want to assign a texture class to every pixel in an image and thus obtain a partitioning of this image. Texture segmentation corresponds to pixelwise texture classification with no a priori knowledge of the number of texture components or the properties of each texture component.

For general reviews on texture analysis the reader is referred to Haralick (1979), van Gool, Dewaele, & Oosterlinck (1985), Tomita & Tsuji (1990), Rao (1990).

1.3 Outline of the thesis

Chapter 2 gives an overview of texture statistics used in texture analysis. This overview is followed by two case studies that evaluates the performance of these statistics in accurately measuring textural properties. First we want to measure the textural changes that cotton textiles undergo during cellulase enzymatic treatment. Then we use second-order statistics for the classification of Brodatz textures.

Chapter 3 deals with parametric description of texture based on a class of models called Markov random fields. The theory of Markov Random fields is reviewed together with the theory of the associated Gibbs random fields. The theory of Gibbs random fields were founded in statistical physics (Ising, 1925) and some relevant results from this area is presented in a new statistical setting. A variety of Markov random fields is reviewed with an emphasis on discrete models. Further we introduce a set of morphological Markov random fields, that extends the standard set of models by using the operators of mathematical morphology (Serra, 1982).

For most practical applications of Markov random fields it is essential that we have accurate and feasible algorithms for parameter estimation. In chapter 4 a selection of estimation methods is reviewed, and some of these methods are applied in chapter 5. An extension of the asymptotic maximum likelihood estimator (Pickard, 1987) to the anisotropic case is proposed in section 4.4.2.

In chapter 5 we review a set of iterative simulation schemes for Markov random field simulation. We then present a fast new parallel algorithm for simulating Markov random fields conditional on given first-order statistics. We investigate the use of this algorithm and a morphological Potts model in the simulation of geological structures.

The Bayesian paradigm is a framework for incorporating stochastic models of visual phenomena into a very general set of tasks from image processing and image analysis. In chapter 6 we give a short review of Bayesian image analysis and present an application that makes successful use of Markov random fields, the Metropolis algorithm and simulated annealing in a Bayesian framework.

Chapter 2

Texture statistics

Texture statistics is frequently classified into first-order, second-order and higher-order statistics. First-order statistics refer to the marginal gray level distribution. Second-order statistics refer to the joint gray level distribution of pairs of pixels and higher-order statistics refer to the joint gray level distribution of three or more pixels.

This chapter gives an overview of texture statistics used in texture analysis. This overview is followed by two case studies that evaluate the performance of these statistics in accurately measuring textural properties. First we want to measure the textural changes that cotton textiles undergo during cellulase enzymatic treatment. Then we use second-order statistics for the classification of Brodatz textures.

2.1 First-order gray level statistics

The first-order gray level statistics can be derived from the *gray level histogram* $\{h_i\}$. h_i is the number of pixels in an image with gray level i. If N is the total number of pixels and G is the number of gray levels then $\sum_{i=0}^{G-1} h_i = N$. The normalized histogram $\{H_i\}$ with $H_i = h_i/N$ is the empirical probability density function for single pixels. Statistics computed from H_i include:

1. The *mean* gray level
$$\mu = \sum_{i=0}^{G-1} iH_i$$
μ measures the average intensity in the image.

2. The gray level *variance*
$$\sigma^2 = \sum_{i=0}^{G-1} (i-\mu)^2 H_i$$
where σ is the *standard deviation*. The variance and the standard deviation measures the global contrast in the image.

3. The *coefficient of variation*
$$cv = \frac{\sigma}{\mu}$$
The coefficient of variation is invariant under a change of scale, $i' = Ai$, thus if the intensity scale has a natural zero, then the cv will be a scale invariant measure of global contrast,

4. The gray level *skewness*
$$\gamma_1 = \frac{1}{\sigma^3} \sum_{i=0}^{G-1} (i-\mu)^3 H_i$$

Skewness measures the extent to which outliers favor one side of the distribution. Skewness is invariant under a linear gray scale transformation $i' = Ai + B$.

5. The gray level *kurtosis*

$$\gamma_2 = \frac{1}{\sigma^4} \sum_{i=0}^{G-1} (i - \mu)^4 H_i - 3$$

Kurtosis measures the peakedness or tail prominence of the distribution. It is 0.0 for the Gaussian distribution. Kurtosis is invariant under a linear gray scale transformation $i' = Ai + B$.

6. The gray level *energy*

$$e = \sum_{i=0}^{G-1} H_i^2$$

where $G^{-1} \leq e \leq 1$. Energy measures the nonuniformity of the histogram.

7. The gray level *entropy*

$$s = -\sum_{i=0}^{G-1} H_i \log H_i$$

where $0 \leq s \leq \log G$. Entropy measures the uniformity of the histogram. This quantity is widely used in image compression. If the logarithm is of base 2, it is the lower bound on the average length of the binary code words used in error-free compression of independent data samples.

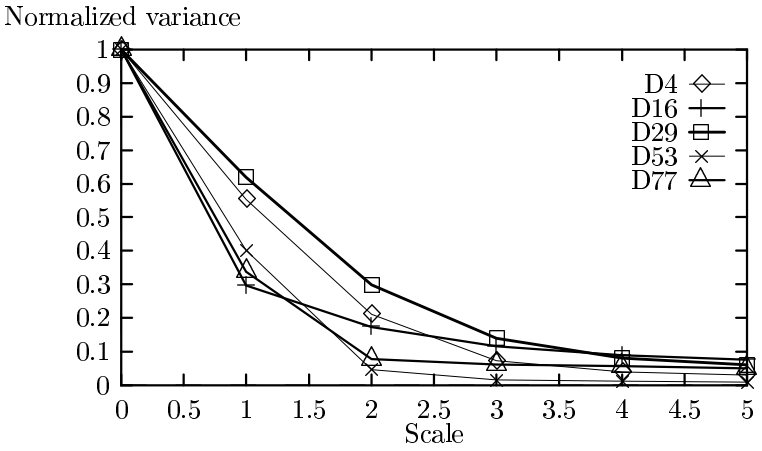

Figure 2.1. Normalized variance versus scale for five Brodatz textures.

2.1.1 Multi-resolution first-order statistics

First-order statistics computed at several different scales (resolutions) will provide us with information about second- and higher-order statistics. As an example we have taken five Brodatz textures, and successively lowpass filtered and subsampled them five times. We have computed the variance of each image and then divided by the variance of the full resolution image. In figure 2.1 we see that the result is five curves, that can be distinguished. Thus multi-resolution first-order statistics contain important textural information.

2.1.2 Histogram matching

The first-order statistics are highly dependent on the lighting conditions. It is therefore common practice to try to eliminate the influence of first-order statistics in texture analysis by making the gray level histogram match a specific distribution. A match to a uniform distribution is called *histogram equalization*, and this is by far the most used match. A match to a Gaussian distribution is another possibility, and in section 2.7 we see that this is a more gentle match especially for stochastic textures.

2.2 Second-order gray level statistics

The autocorrelation (or the closely related variogram) is probably the best known second-order gray level statistic. We will, however, consider the second-order gray level statistics in a more general setting: *gray level cooccurrence matrices (GLCM)*. The autocorrelation can be computed from these cooccurrence matrices.

2.2.1 Gray level cooccurrence matrices

The gray level cooccurrence matrices are a full representation of the second-order gray level statistics. A GLCM, $\mathbf{c}$, is defined with respect to a given (row,column) displacement $\mathbf{h}$, and element (i,j), denoted c_{ij}, is the number of times a point having gray level j occurs in position $\mathbf{h}$ relative to a point having gray level i. Let $N_{\mathbf{h}}$ be the total number of pairs, then $C_{ij} = c_{ij}/N_{\mathbf{h}}$ denotes the elements of the normalized GLCM, $\mathbf{C}$.

The meaning of the above definition gets more apparent if we as an example compute $\mathbf{c}$ from the 4-color image

2	1	1	3	3
3	1	0	2	2
3	3	1	1	2
0	2	1	0	1
0	0	1	0	0

If $\mathbf{h}$=(0,1), i.e. one step in the horizontal direction, then $\mathbf{c}$ will be

	0	1	2	3
0	2	2	2	0
1	3	2	1	1
2	0	2	1	0
3	0	2	0	2

and $N_{\mathbf{h}}$ will be equal to 20.

It is easily seen that

$$\mathbf{C}(-\mathbf{h}) = \mathbf{C}^T(\mathbf{h})$$

where $\mathbf{C}^T$ is $\mathbf{C}$ transposed.

A symmetric GLCM, $\mathbf{c}_s(\mathbf{h})$, can be obtained by pooling the frequencies of $\mathbf{c}(\mathbf{h})$ and $\mathbf{c}(-\mathbf{h})$. Hence

$$\mathbf{c}_s(\mathbf{h}) = \mathbf{c}(\mathbf{h}) + \mathbf{c}^T(\mathbf{h})$$

and

$$\mathbf{C}_s(\mathbf{h}) = \frac{1}{2}[\mathbf{C}(\mathbf{h}) + \mathbf{C}^T(\mathbf{h})]$$

Assuming isotropy (no directionality) we can pool the frequencies of cooccurrence matrices with displacements $\mathbf{h}$ of different angles and approximately the same length h. This provides us with the isotropic GLCM, $\mathbf{c}_i(\mathbf{h})$, where

$$\mathbf{c}_i(1) = \mathbf{c}_s(0,1) + \mathbf{c}_s(1,0) + \mathbf{c}_s(1,1) + \mathbf{c}_s(-1,1)$$

and

$$\mathbf{C}_i(1) = \frac{1}{4}[\mathbf{C}_s(0,1) + \mathbf{C}_s(1,0) + \mathbf{C}_s(1,1) + \mathbf{C}_s(-1,1)]$$

One of the main problems associated with the use of cooccurrence matrices is that they have to be computed for many different values of $\mathbf{h}$, thus providing us with an immense amount of data. Data reduction can be accomplished by pooling the matrices as shown above, by reducing the number of gray levels or by computing texture features from each matrix. These features can then be used for description and classification of textures.

Let

$$C_i^x = \sum_{j=0}^{G-1} C_{ij}$$

$$C_j^y = \sum_{i=0}^{G-1} C_{ij}$$

and let μ_x, μ_y, σ_x and σ_y be the means and standard deviations of C_i^x and C_j^y over i and j. Then a number of features can be computed from the GLCM including:

1. Energy or Angular Second Moment

$$\mathcal{E} = \sum_{i=0}^{G-1}\sum_{j=0}^{G-1} C_{ij}^2$$

where $G^{-2} \leq \mathcal{E} \leq 1$. $\mathcal{E}$ takes the value G^{-2} for a uniform distribution over $\mathbf{C}$, and the value 1 iff only one cell is nonzero.

2. Entropy

$$\mathcal{S} = -\sum_{i=0}^{G-1}\sum_{j=0}^{G-1} C_{ij} \log C_{ij}$$

where $0 \leq \mathcal{S} \leq \log G^2$. $\mathcal{S}$ takes the value $\log G^2$ for a uniform distribution over $\mathbf{C}$, and the value 0 iff only one cell is nonzero.

3. Maximum Probability

$$\mathcal{M} = \max C_{ij}$$

where $G^{-2} \leq \mathcal{M} \leq 1$. $\mathcal{M}$ takes the value G^{-2} for a uniform distribution over $\mathbf{C}$, and the value 1 iff only one cell is nonzero.

4. Correlation (or Autocorrelation)

$$\rho = \sum_{i=0}^{G-1}\sum_{j=0}^{G-1} \frac{(i-\mu_x)(j-\mu_y)C_{ij}}{\sigma_x\sigma_y}$$

where $-1 \leq \rho \leq 1$. ρ takes the value 1 iff only values on the main diagonal of $\mathbf{C}$ are nonzero and the value 0 iff the gray values are uncorrelated.

5. Diagonal Moment

$$\mathcal{D} = \sum_{i=0}^{G-1}\sum_{j=0}^{G-1} |i-j|(i+j-\mu_x-\mu_y)C_{ij}$$

The diagonal moment basically measures the difference in correlation for high gray levels and for low gray levels. It is mentioned in Laws (1980), but has otherwise been left out in most studies of GLCM.

6. Informational Coefficient of Correlation

$$r_1 = \sqrt{1 - e^{-2r_0}}$$

where

$$r_0 = -\sum_{i=0}^{G-1}\sum_{j=0}^{G-1} C_i^x C_j^y \log(C_i^x C_j^y) - \mathcal{S}$$

is the Logarithmic Index of Correlation (Linfoot, 1957). Since $r_0 \geq 0$ we have $0 \leq r_1 < 1$.

7. Maximal Correlation Coefficient. This feature is the square root of the second largest eigenvalue of $\mathbf{Q}$ where

$$Q_{ij} = \sum_k \frac{C_{ik}C_{jk}}{C_i^x C_k^y}$$

Let $\mathbf{R}$ and $\mathbf{S}$ be equal to $\mathbf{C}$ with respectively row sums and column sums normalized to unity, i.e.

$$R_{ij} = \frac{C_{ij}}{C_i^x}$$

$$S_{ij} = \frac{C_{ij}}{C_j^y}$$

then

$$\mathbf{Q} = \mathbf{R}\mathbf{S}^T$$

$\mathbf{R}$, $\mathbf{S}^T$ and $\mathbf{Q}$ are stochastic matrices, i.e. their largest eigenvalue is 1. If they are considered as transition matrices for a Markov chain and if they are irreducible then the histogram vector $\mathbf{p}$ will be the unique invariant distribution for the Markov chain. The rate of convergence to the invariant distribution is determined by the second-largest eigenvalue, λ_2, where $0 \leq \lambda_2 < 1$ (Seneta, 1981). $\mathbf{Q}$ is the transition matrix for one jump with displacement vector $\mathbf{h}$ and back again. If we start at one pixel with an initial gray level distribution and then makes successive jumps back and forth, then the gray level distribution will approach the invariant gray level distribution. The memory of the gray level distribution retained in each jump back and forth is determined by the second-largest eigenvalue, λ_2, of $\mathbf{Q}$. If the pixels on each side of a jump are independent, we have $\lambda_2 = 0$.

Energy, entropy and maximum probability are uniformity measures. They all have one extremum for the uniform distribution and another extremum when one probability equals unity. The difference between these measures is demonstrated for two distributions with 4 possible outcomes.

p_1	p_2	p_3	p_4	Energy	Entropy
0.50	0.50	0.00	0.00	0.50 (1)	0.69 (2)
0.76	0.08	0.08	0.08	0.60 (2)	0.81 (1)

The uniformity rankings are shown in parentheses. The energy measure assumes the first distribution to be the most uniform of the two, while the entropy measure chooses the second. We see that, when measuring uniformity, energy penalizes single high probabilities, while entropy penalizes zero probabilities. If we increase the zero probabilities of the first distribution a little then entropy will reverse the ranking, and make this the most uniform distribution according to both measures.

p_1	p_2	p_3	p_4	Energy	Entropy
0.48	0.48	0.02	0.02	0.46 (1)	0.86 (1)

Maximum probability measures uniformity solely on the basis of the highest probability and the ranking by this measure will often agree with that based on energy.

The cooccurrence matrix itself can also be used as a feature (Vickers & Modestino, 1982; Parkkinen & Oja, 1986).

The use of GLCM in texture analysis is sometimes referred to as the *spatial gray level dependence method (SGLDM)*.

2.2.2 Gray level difference histogram

The *gray level difference histogram (GLDH)* is a histogram of the absolute differences of gray levels from pairs of pixels. It is computed from the GLCM by summing the two-dimensional density C_{ij} over constant value of $|i-j|$. The GLDH can be regarded as a histogram of the "distance" to the main diagonal in the GLCM.

$$D_k = \underbrace{\sum_{i=0}^{G-1}\sum_{j=0}^{G-1}}_{|i-j|=k} C_{ij}, k = 0,..,G-1$$

The features computed from the GLDH include:

1. Difference Energy
$$\mathcal{DE} = \sum_{k=0}^{G-1} D_k^2$$
where $G^{-1} \leq \mathcal{DE} \leq 1$.

2. Difference Entropy
$$\mathcal{DS} = -\sum_{k=0}^{G-1} D_k \log D_k$$
where $0 \leq \mathcal{DS} \leq \log G$.

3. Inertia, Contrast or Variogram
$$\mathcal{I} = \sum_{k=0}^{G-1} k^2 D_k = 2\sigma^2(1-\rho) \quad (2.1)$$
where σ is the gray level variance and ρ is the correlation.

4. Inverse Difference Moment or Local Homogeneity
$$\mathcal{IDM} = \sum_{k=0}^{G-1} \frac{D_k}{1+k^2}$$

5. Difference Variance
$$\mathcal{DV} = \mathcal{I} - (\sum_{k=0}^{G-1} kD_k)^2$$

GLDH features are a subset of GLCM features, and this relation will subsequently be implicit. The only useful way of comparing the two sets of features is to determine the loss of information when going from GLCM to GLDH. Feature computation from the GLDH is often called the *gray level difference method (GLDM)*. The advantage of GLDM is the lower storage requirements and lower computational complexity.

2.2.3 Gray level sum histogram

The *gray level sum histogram (GLSH)* is a histogram of the sum of pairs of pixels. It is computed from the GLCM by summing the two-dimensional

density C_{ij} over constant value of $(i+j)$, i.e.

$$S_k = \underbrace{\sum_{i=0}^{G-1}\sum_{j=0}^{G-1}}_{i+j=k} C_{ij}, k = 0,..,2G-2$$

We will use the Sum Average below

$$\mathcal{SA} = \sum_{k=0}^{2G-2} kS_k = \mu_x + \mu_y$$

The features computed from the GLSH include:

1. Sum Energy
$$\mathcal{SE} = \sum_{k=0}^{2G-2} S_k^2$$
where $(2G-1)^{-1} \leq \mathcal{SE} \leq 1$.

2. Sum Entropy
$$\mathcal{SS} = -\sum_{k=0}^{2G-2} S_k \log S_k$$
where $0 \leq \mathcal{SS} \leq \log(2G-1)$.

3. Sum Variance
$$\mathcal{SV} = \sum_{k=0}^{2G-2} (k-\mathcal{SA})^2 S_k = 2\sigma^2(1+\rho)$$
where σ is the gray level variance and ρ is the correlation.

4. Cluster Shade
$$\mathcal{A} = \sum_{k=0}^{2G-2} (k-\mathcal{SA})^3 S_k$$

5. Cluster Prominence
$$\mathcal{B} = \sum_{k=0}^{2G-2} (k-\mathcal{SA})^4 S_k$$

GLSH-features has not been used as widely as features based on GLDH. Conners, Trivedi, & Harlow (1984) found that cluster shade and cluster prominence was a useful supplement to the GLCM and GLDH features mentioned above. Like the GLDH, the GLSH has lower storage requirements and lower computational complexity, but no authors have to our knowledge tried the GLSH features by themselves.

It is obvious that the GLDH and GLSH contains a lot of the information from the cooccurrence matrices. Due to the lower computational complexity it is, as mentioned for the GLDH, relevant to investigate if any significant information is lost when going from the GLCM to the GLDH and GLSH. The structure of the GLCM is diagonal and often at least approximately symmetric. Unser (1986b) approximated the energy and entropy features of the GLCM from the GLDH and the GLSH. This approximation gave only a slight decrease in classification accuracy. Unser (1986b) noted that the sum and difference of pairs of pixels are decorrelated, but this is not generally true for the sum and the absolute difference. The diagonal moment measures this correlation and it varies from texture to texture (See section 2.7).

Unser (1986b) also used the GLDH and GLSH themselves as features.

2.2.4 Haralick features

Most of the features mentioned in this section were introduced in texture analysis in a paper by Haralick, Shanmugam, & Dinstein (1973), where 14 different features (f1-f14) were presented. They are all natural descriptors of two-dimensional distributions, although they seem to have been selected in a rather ad hoc manner. Even though it is recognized that these features do not describe all aspects of the cooccurrence matrices they have been used very rigorously in many papers. Three of the features were not included in the list of GLCM features.

- Variance (f4). The gray level variance belongs to the first-order statistics.
- Sum Average (f6)

$$f_6 = \mathcal{SA} = \sum_{k=0}^{2G-2} kS_k = \mu_x + \mu_y$$

  This feature also belongs to the first-order statistics.

- Information Measures of Correlation (f12 and f13).

$$f_{12} = \frac{HXY - HXY1}{\max(HX, HY)}$$

$$f_{13} = \sqrt{1 - \exp(-2(HXY2 - HXY))}$$

  where $HXY = \mathcal{S}$ and

$$HXY1 = -\sum_{i=0}^{G-1}\sum_{j=0}^{G-1} C_{ij}\log(C_i^x C_j^y)$$

$$HXY2 = -\sum_{i=0}^{G-1}\sum_{j=0}^{G-1} C_i^x C_j^y \log(C_i^x C_j^y)$$

As mentioned in Linfoot (1957) $HXY1 = HXY2$. f12 and f13 are thus closely related, and only f13, the informational coefficient of correlation, is considered.

2.2.5 GLCM as a contingency table

Zucker & Terzopoulos (1980) interpreted the cooccurrence matrix $\mathbf{C}$ as a normalized contingency table (See e.g. Bishop, Fienberg, & Holland (1975)) and used the χ^2 statistic to select matrices suitable for classification.

$$\chi^2 = N_{\mathbf{h}} \sum_{i=0}^{G-1}\sum_{j=0}^{G-1} \frac{(C_{ij} - C_i^x C_j^y)^2}{C_i^x C_j^y} = N_{\mathbf{h}}\left(\sum_{i=0}^{G-1}\sum_{j=0}^{G-1} \frac{C_{ij}^2}{C_i^x C_j^y} - 1\right)$$

The χ^2 values and the selected displacements $\mathbf{h}$ can be used as features for classification.

The χ^2 statistic measures the association between variables in contingency tables, but does not discriminate among the types of association. Figueiras-Vidal, Paez-Borrallo, & Garcia-Gomez (1987) pointed out that periodicity is indicated in a cooccurrence matrix by a concentration of high counts around the main diagonal. They suggested the inertia measure (2.1) to detect periodicities.

The κ measure of agreement (Cohen, 1960)

$$\kappa = \frac{\sum_{i=0}^{G-1} C_{ii} - \sum_{i=0}^{G-1} C_i^x C_i^y}{1 - \sum_{i=0}^{G-1} C_i^x C_i^y}$$

was suggested by Parkkinen, Selkäinaho, & Oja (1990) to detect periodicities. It directly measures the concentration on the main diagonal and the

computational complexity is $O(G)$ instead of $O(G^2)$ for the χ^2 statistic. The κ statistic works best with a limited number of gray levels, e.g. from 4 to 32, and, as we will see in section 2.4, it corresponds to the correlation measure for binary textures.

Many other features can be used to select the displacement(s) that give the best classification.

2.2.6 Multi-resolution GLCM

Weszka, Dyer, & Rosenfeld (1976) concluded that large-distance cooccurrence features gave better performance if a spatial averaging was done first. This suggests that cooccurrence matrices at several different scales should be considered.

2.2.7 GLCM performance

GLCM features have an extensive history as a reference for texture feature performance. We shall give a brief summary.

Haralick et al. (1973) used GLCM features to classify photomicrographs of sandstone, panchromatic aerial photographs and multispectral satellite imagery. They found that textural features is a valuable supplement to spectral features.

Weszka et al. (1976) compared the classification performance on aerial photographs and LANDSAT imagery of GLCM features, GLDH features, ring

and wedge features in the spatial frequency domain and gray level run length features. GLCM and GLDH features were found to be the most useful and of almost equal performance.

Conners & Harlow (1980) made a theoretical comparison of the same groups of features, and the results agree very well with those of Weszka et al. (1976). Many authors have since then introduced new textural features and claimed these to be superior to the GLCM features.

Laws (1980) claimed that his texture energy features performed significantly better than GLCM features in segmentation of a composite of eight Brodatz textures.

Kashyap, Chellappa, & Khotanzad (1982) use the maximum likelihood estimate of the parameters in a Simultaneous Autoregressive Model (SAR) as features for classification. The result is comparable to that of GLCM features.

Vickers & Modestino (1982) used an isotropic cooccurrence matrix to classify subimages of 9 Brodatz textures. For distances of 1,3 and 5 they obtained between 95% and 98% correctly classified. Parkkinen & Oja (1986) used cooccurrence matrices with a horizontal displacement.

Siew, Hodgson, & Wood (1988) used GLCM, GLDH, gray level run length and neighboring gray level dependence features to measure carpet wear. Their distinction between GLCM and GLDH features is noninformative since 2 features are common to both groups, and they use a standard GLCM and an isotropic GLDH. The result shows that features from all four groups can characterize the appearance of carpets. Their research indicates that

these statistical measures are superior to a trained panel in reliably ranking carpets according to wear.

du Buf, Kardan, & Spann (1990) compared 7 sets of features and found that GLCM, Laws (Laws, 1980) and Unser (Unser, 1986a) features were generally best.

Berry & Goutsias (1991) made a comparison between features based on the neighboring gray level dependence matrix (NGLDM) of Sun & Wee (1983) and GLCM features. On synthetic textures NGLDM features performed better. On natural textures they performed equally well.

2.3 Higher-order gray level statistics

Higher-order gray level statistics were declared unimportant for texture perception by the now disproved iso-second-order conjecture (Julesz, 1975), but they seem to have regained their popularity in the literature. We will review a few approaches.

2.3.1 Gray level run length matrix

A gray level run is a set of consecutive, collinear pixels with the same gray level. The number of pixels in a run is the run length. Galloway (1975) used a *gray level run length matrix (GLRLM)* to compute texture features. Element (i,j) of the GLRLM, **r**, is denoted r_{ij}, and this is the number of runs of gray level i having length j. The total number of runs is N_r. The

GLRLM can be computed in any direction, but usually only directions 0°, 45°, 90° and 135° are used. From the image

2	1	1	3	3
3	1	0	2	2
3	3	1	1	2
0	2	1	0	1
0	0	1	0	0

we can compute **r** for the horizontal direction (0°).

GLRLM	1	2	3	4	5
0	3	2	0	0	0
1	4	2	0	0	0
2	3	1	0	0	0
3	1	2	0	0	0

where $N_r = 18$.

The following features, computed from the GLRLM, were suggested:

1. Short Runs Emphasis

$$RF1 = \sum_{i=0}^{G-1} \sum_{j=1}^{L} \frac{R_{ij}}{j^2}$$

where $G^{-2} \leq \mathcal{E} \leq 1$. $\mathcal{E}$ takes the value G^{-2} for a uniform distribution of the counts and the value 1 iff only one cell is nonzero.

2. Long Runs Emphasis

$$RF2 = \sum_{i=0}^{G-1} \sum_{j=1}^{L} j^2 R_{ij}$$

3. Gray Level Nonuniformity

$$RF3 = \sum_{i=0}^{G-1} [\sum_{j=1}^{L} R_{ij}]^2$$

where $1/G \leq RF3 \leq 1$. $RF3$ takes the value $1/G$ for a uniform distribution of the counts and the value 1 iff only one cell is nonzero.

4. Run Length Nonuniformity

$$RF4 = \sum_{j=1}^{L} [\sum_{i=0}^{G-1} R_{ij}]^2$$

where $1/L \leq RF4 \leq 1$. $RF4$ takes the value $1/L$ for a uniform distribution of the counts and the value 1 iff only one cell is nonzero.

5. Run Percentage

$$RF5 = N_r/N$$

where $1/N \leq RF5 \leq 1$.

The GLRLM features are very sensitive to noise, and this is probably the reason for the reported bad performance (e.g. Weszka et al. (1976)). The performance for discrete (e.g. binary) textures is likely to be better.

2.3.2 Neighboring gray level dependence matrix

The *neighboring gray level dependence matrix (NGLDM)* was introduced by Sun & Wee (1983). In this approach all neighbors of a pixel are considered

at the same time. A neighbor is a pixel within a certain distance d of the central pixel and S is the number of neighbors. d is usually chosen to be $\sqrt{2}$ and then $S = 8$. A pixel and its neighbor are said to have similar gray levels if the absolute gray level difference is less than or equal to a chosen positive number a. Element (k,s) of a NGLDM, $\mathbf{q}$, is denoted q_{ks}, and this is the number of pixels with gray level k having s neighbors with similar gray levels. Let N_d be the total number of counts in $\mathbf{q}$, then $\mathbf{Q} = \mathbf{q}/N_d$ is the normalized NGLDM. The notation presented here differs from the notation of Sun and Wee. This is to keep the definitions along the lines of the GLCM definition. From the image

2	1	1	3	3
3	1	0	2	2
3	3	1	1	2
0	2	1	0	1
0	0	1	0	0

we can compute $\mathbf{q}$ for $a = 0$ and $d = \sqrt{2}$ as

NGLDM	0	1	2	3	4	5	6	7	8
0	1	0	1	0	0	0	0	0	0
1	0	0	0	4	0	0	0	0	0
2	1	0	1	0	0	0	0	0	0
3	0	0	1	0	0	0	0	0	0

where $N_d = 9$.

The features, that Sun and Wee suggest computed from the NGLDM, are listed below with the modification that the feature computation here is

based on the normalized NGLDM. This means that the features are independent of N_d.

1. Small Number Emphasis

$$N1 = \sum_{k=0}^{G-1} \sum_{s=0}^{S} \frac{Q_{ks}}{1+s^2}$$

where $G^{-2} \leq \mathcal{E} \leq 1$. $\mathcal{E}$ takes the value G^{-2} for a uniform distribution of the counts and the value 1 iff only one cell is nonzero.

2. Large Number Emphasis

$$N2 = \sum_{k=0}^{G-1} \sum_{s=0}^{S} s^2 Q_{ks}$$

3. Number Nonuniformity

$$N3 = \sum_{s=0}^{S} [\sum_{k=0}^{G-1} Q_{ks}]^2$$

4. Second Moment

$$N4 = \sum_{k=0}^{G-1} \sum_{s=0}^{S} Q_{ks}^2$$

5. Entropy

$$N5 = -\sum_{k=0}^{G-1} \sum_{s=0}^{S} Q_{ks} \log Q_{ks}$$

2.4 Statistics for binary images

For binary images the number of gray levels, G, equals 2 and the first-order statistics are determined by the fraction of 1-pixels, $p_1 = n_1/N$. We have:

$$\mu = p_1$$

$$\sigma^2 = p_1(1-p_1)$$

A GLCM has the form:

GLCM	0	1	sum
0	n_{00}	n_{01}	$n_{0.}$
1	n_{10}	n_{11}	$n_{1.}$
sum	$n_{.0}$	$n_{.1}$	$N_{\mathbf{h}}$

The normalized version is:

GLCM	0	1	sum
0	p_{00}	p_{01}	$p_{0.}$
1	p_{10}	p_{11}	$p_{1.}$
sum	$p_{.0}$	$p_{.1}$	1

For stationary images we have $p_{.0} \approx p_{0.} \approx p_0$, $p_{.1} \approx p_{1.} \approx p_1$ and

$$p_{01} \approx p_{10} \approx p_1 - p_{11}$$

$$p_{00} \approx p_0 - p_1 + p_{11}$$

We see, that, given p_1, all second-order statistics can be expressed as a function of e.g. p_{11}, i.e. there is only 1 degree of freedom in a GLCM given the first-order statistic p_1.

It is more instructive to see the normalized GLCM expressed in terms of the first-order statistics and the correlation $\rho = \frac{p_{11}-p_1 p_1}{p_0 p_1}$.

GLCM	0	1	sum
0	$p_0 p_0(1+\frac{p_1}{p_0}\rho)$	$p_0 p_1(1-\rho)$	p_0
1	$p_1 p_0(1-\rho)$	$p_1 p_1(1+\frac{p_0}{p_1}\rho)$	p_1
sum	p_0	p_1	1

Thus all second-order statistics can be expressed in terms of the first-order statistics and the correlation ρ. We shall show this for the χ^2 measure and the κ measure.

The χ^2 measure is for binary textures

$$\chi^2 = N_{\mathbf{h}}[\frac{(p_{00}-p_0 p_0)^2}{p_0 p_0} + 2\frac{(p_{01}-p_0 p_1)^2}{p_0 p_1} + \frac{(p_{11}-p_1 p_1)^2}{p_1 p_1}]$$

$$= N_{\mathbf{h}}[p_0^2\rho^2 + 2p_0 p_1\rho^2 + p_1^2\rho^2] = N_{\mathbf{h}}[\rho^2(p_0+p_1)^2] = N_{\mathbf{h}}\rho^2$$

and the κ measure is

$$\kappa = \frac{p_{00}+p_{11}-p_0 p_0 - p_1 p_1}{1-p_0 p_0 - p_1 p_1} = \frac{2p_0 p_1\rho}{2p_0 p_1} = \rho$$

Higher-order statistics are elegantly expressed in terms of mathematical morphology as in Serra (1982).

2.5 Fourier features

The discrete Fourier transform (DFT), $\mathcal{F}$, and its inverse, $\mathcal{F}^{-1}$, are defined for the image, $\{f(m,n), m=0,..,M-1, n=0,..,N-1\}$, as

$$\mathcal{F}(f) = F(u,v) = \frac{1}{MN}\sum_{m=0}^{M-1}\sum_{n=0}^{N-1} f(m,n)e^{-j2\pi(\frac{mu}{M}+\frac{nv}{N})}$$

and

$$\mathcal{F}^{-1}(F) = f(m,n) = \sum_{u=0}^{M-1}\sum_{v=0}^{N-1} F(u,v)e^{j2\pi(\frac{mu}{M}+\frac{nv}{N})}$$

The Fourier power spectrum is

$$|F|^2 = FF^* \tag{2.2}$$

where F^* denotes the complex conjugate of F. The power spectrum usually varies over several orders of magnitude, which makes it interesting to consider the log-power spectrum

$$log\,(1+\mid F\mid^2) \tag{2.3}$$

Standard library FFT routines usually have the restriction that the height and the width of the image has to be a power of 2.

The power spectrum is the Fourier transform of the autocorrelation, i.e. it only contains information about the second-order statistics. It is recognized that the phase spectrum contains much relevant information, but it is very hard to make it useful.

From the Fourier transform of the image, the power spectrum and the log-power spectrum we can compute a number of features. Averages of the power spectrum over ring-shaped and wedge-shaped regions are common

features (see e.g. Weszka et al. (1976)). Liu & Jernigan (1990) extracts 28 features from the power and phase spectrum.

2.6 Measurement of enzymatic treatment effect on textile

The effect of cellulase enzymatic treatment on textiles has been investigated using standard texture algorithms. An extensive study in both the Fourier domain and the spatial domain has revealed the nature of the changes and resulted in one single feature that measures these changes in a fast and robust way.

2.6.1 Background

This project started when the R&D group in the detergent enzyme division of Novo Nordisk (a world-leading manufacturer of detergent enzymes) expressed the wish to quantify the effects of enzymatic treatment of textiles using digital image analysis. Until now this quantification has been done qualitatively using microscopic inspection and quantitatively using panel tests and light measurements (Hunter coordinates). There was a need for a new objective, robust, fast and relatively inexpensive method.

2.6.2 Image acquisition

The image acquisition is carried out as follows. The textile is placed in homogeneous and plentiful lighting. A camera is positioned such that it's optical axis is perpendicular to the textile plane and the rectangular visual

area covers as much of the textile as possible without including non-textile areas. The size of the textiles in this study is 15x10cm.

We used an RGB high-resolution slow-scan camera. The camera output is digitized by a framegrabber that generates frames of 978 by 768 pixels in the red, green and blue band. These frames are cut to 969 by 711 to eliminate acquisition artifacts. Subsequently we will only show results derived from the green band since the textiles used in this experiment are black and gray and thus contains very little or no color information.

This study regards the enzymatic treatment effect for a single type of cellulase. We want to assess the effect at different pH values and for different doses. To assess the day-to-day variation the textiles were washed on different days for each pH-level. Thus we have three factors that we want to investigate.

- pH: 3 levels, 1 2 3
- dose: 8 levels, 0 10 25 40 50 75 100 200
- day(pH): 3 levels, 1 2 3 (for pH values 7.0,8.0 and 9.0)

We have two repetitions for each combination, thus we end up with 144 images. In figure 2.2 we see 8 textiles representing the 8 doses for pH 1, day 1 and repetition 1.

Figure 2.2. 8 textiles representing the 8 doses for pH 1, day 1.

2.6.3 Description of visual properties

The object of the digital image analysis is to compute one feature that quantifies a given visual property from the image array. In this case the visual property is the human perception of wear. The feature has to correlate well with panel tests. For cellulase enzymatic treatment with known effects this means that the feature has to show improvement as a function of dose and show best results for pH values close to the pH with highest enzyme activity (between 7.0 and 8.0 in our case).

Obviously many different features can be computed from the image. A simple feature is the average intensity, *lightness*. This has a strong resemblance to what is measured by the Hunter coordinates. Probably this lightness feature also has a strong influence on a panel test. Figure 2.3 shows the average intensity as a function of dose for pH level 1. We see that lightness only has discriminative capability for small doses. In the context of image analysis lightness is a non-robust feature in the sense that it depends heavily on lighting conditions and camera sensitivity.

Another aspect of enzymatic effect on the textiles estimated by the panel test is the *distinctness* of the regular textile pattern. This distinctness should increase as a result of the cellulase enzymatic treatment. The regular pattern in the investigated textiles resembles a rectangular grid structure. The well defined period of this grid makes it appropriate to look at the textiles in the Fourier domain. This is done in the next section.

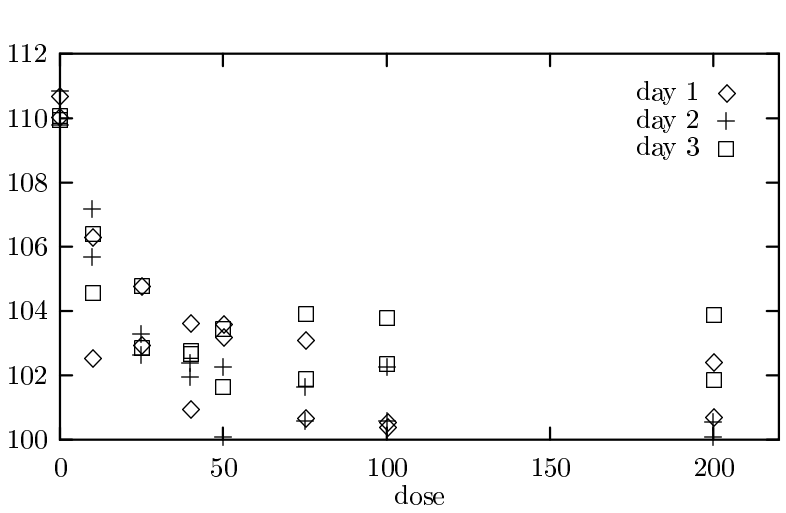

Figure 2.3. Average intensity as a function of dose for pH level 1.

2.6.4 Analysis in the Fourier domain

Frequency based methods.

The classical way of obtaining an estimate of the power spectrum is by equation 2.2. The log-power spectrum is given in equation 2.3.

The periodogram is a non-consistent estimate of the power spectrum. *Welch's method* is one way to deal with this. The image is split up in a number of non-overlapping subimages. The periodogram is calculated in each subimage, followed by an averaging over the subspectra.

Figure 2.4 shows the full resolution power spectra of the textiles in figure 2.2. The concentric circles are isolines for the spatial frequency. Several high-intensity spots in the power spectrum is showing the periodicity of the weaves. The spots of lower intensity in the high-frequency areas are higher harmonics. We see that the intensity in the low-frequency areas (near the center of the power spectrum) is fading for higher doses of enzyme. To illustrate this effect we computed the average of the power spectrum in the rings between the concentric circles and plotted it versus the radius of the rings. These averages are computed for each of the power spectra in figure 2.4, and the average corresponding to dose 0 subtracted from the averages of each of the other doses. The plot is shown in figure 2.5, and it is obvious that the averages in the low-frequency areas are decreasing for higher doses. We also note that all the curves has approximately the same intersection at a frequency corresponding to the frequency of the weaves. Thus having established that the power spectrum actually contains relevant information about the textile wear, we will try to quantify this in a single Fourier feature.

Figure 2.4. Power spectra of the textiles in figure 2.2.

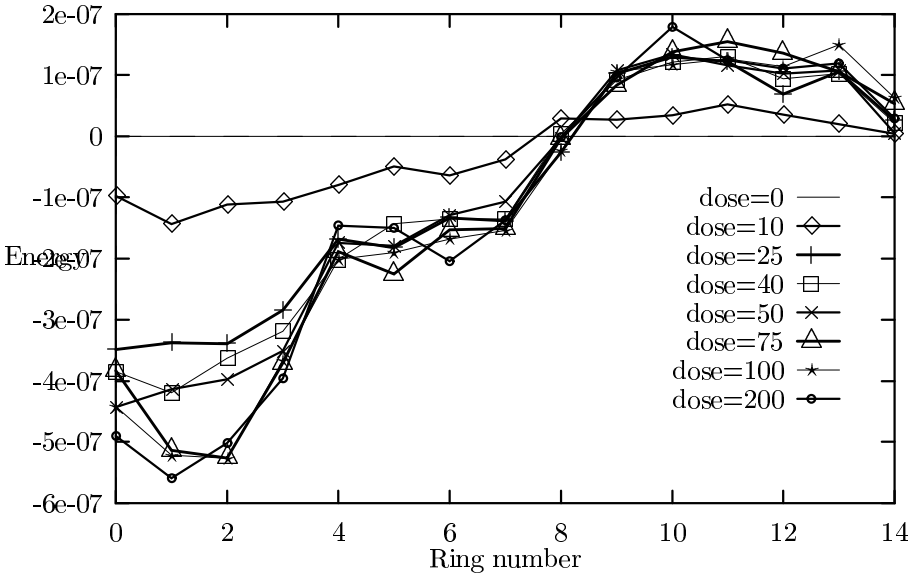

Figure 2.5. Average of power spectra rings relative to dose 0 for the spectra in figure 2.4.

Spectral texture features.

Texture features derived in the spatial frequency domain have been investigated e.g. in Weszka et al. (1976) and Liu & Jernigan (1990). The features tested in the present context are listed below.

1. Rings
2. Wedges
3. Inertia
4. Entropy
5. Anisotropy

All features has been computed on both the full resolution power spectrum and the power spectrum estimated using Welch's method. The Welch spectral estimate performed significantly better than the full resolution power spectrum.

The features were computed on both the power spectrum and the log-power spectrum. It turned out that the features calculated on the log-power spectrum performed significantly better than the power spectral features. Furthermore we found that inertia and entropy features performed better than the other features. The inertia feature performed generally a little better than the entropy feature, and it seems to be a more natural way summarize the phenomena observed in figure 2.5.

The inertia feature I and log-power inertia LI is computed as

$$I = \sum_{(u,v)} (u^2 + v^2) \mid F(u,v) \mid^2$$

$$LI = \sum_{(u,v)} (u^2 + v^2) \log(1 + \mid F(u,v) \mid^2)$$

where we are summing over all frequencies. The normalized inertia is the inertia divided by the inertia of the corresponding textile with $dose = 0$. In figure 2.6 we show the normalized log-power inertia vs. log(dose) for all three values of pH. Thus the measure is averaged over days and repetitions. It can be seen that there is a clear distinction between the performance of the enzymes at the three pH values. In addition there seems to be an approximately linear relation between the inertia and log(dose).

Discussion of results

The spectral approach has provided us with a useful feature and a lot of insight regarding the nature of this problem. The use of the FFT algorithm however introduces some, somewhat technical, limitations regarding the size of the image and computational speed. It is also less flexible in removing textile irregularities from the analysis.

2.6.5 Spatial domain features

The distinctness property and other textural properties can also be measured by textural features in the spatial domain. Siew et al. (1988) used features based on different texture matrices for carpet wear assessment. The

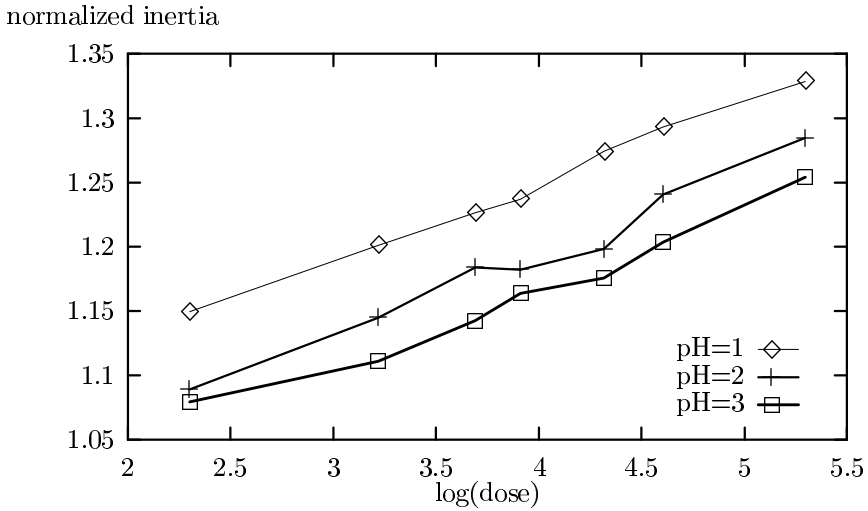

Figure 2.6. Normalized log-power inertia versus log(dose). We see that the measure reflects the expected ranking.

conclusion of the paper was, that features based on texture matrices (e.g. GLCM) can be used to characterize the appearance of carpets and changes, they undergo during wear. The problem of carpet wear assessment is similar to measuring effects of enzymatic treatment, and therefore we included GLCM features in our study.

Spatial features

The spatial domain features included in this study were all the first-order statistics of section 2.1 and the following 15 GLCM features.

1. Energy
2. Entropy
3. Maximum probability
4. Correlation
5. Diagonal correlation
6. Kappa
7. Difference energy
8. Difference entropy
9. Inertia
10. Local homogeneity
11. Sum energy

12. Sum entropy
13. Sum variance
14. Cluster shade
15. Cluster prominence

The features were computed for several numbers of gray levels and at several resolutions. Attempts to make the features robust have included correction for inhomogeneous lighting and automatic removal of textile irregularities.

The operational feature

Many of the tested features performed well on subsets of the images, but only a few features gave an overall good and robust measurement.

It was possible to find a relatively simple feature with an overall good and robust performance. This feature is computed as follows. The image is transformed to a resolution where the regular textile pattern has just disappeared (in our case the images were lowpass-filtered and subsampled to 1/16 size). Then the variance of this image is computed. The variances are normalized (divided) by the variance of the corresponding textile with dose=0. The average over days and repetitions of this feature is shown in figure 2.7 in a log-log plot. It ranks the textiles just as expected and it seems that a linear fit is appropriate for each pH level. This feature shall be called the *coarse-scale normalized variance (csnv) feature*. The csnv feature can be compared to the the Fourier inertia feature in the Fourier domain. The lowpass filter we used correspond approximately to a multiplication

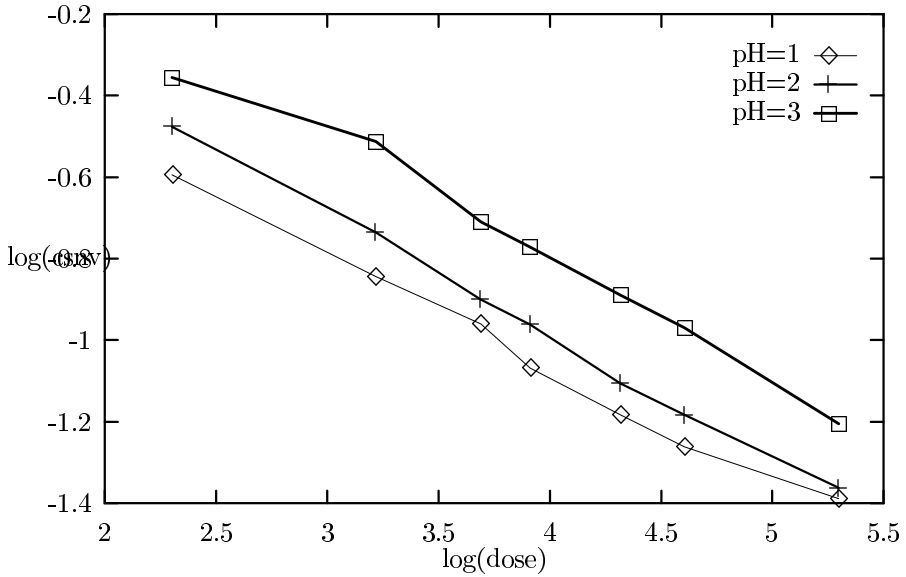

Figure 2.7. Plot of (log) coarse-scale normalized variance versus log(dose). We see that the measure reflects the expected ranking.

with a Gaussian weighting function centered at (0,0) in the Fourier domain. For the Fourier inertia feature the weighting function is $(u^2 + v^2)$. Thus the csnv feature measures the energy in the low frequencies and the inertia feature measures the energy in the high frequencies. Since the measures are normalized they will actually measure similar properties, but as the textile wear seems to be best described in the low frequencies, the inertia feature is not as robust as the csnv feature.

Fitting a general linear model with the SAS GLM-procedure:

```
proc glm;
  class ph day;
  model logvar = logdose ph day(ph) ph*logdose logdose*day(ph) ;
  lsmeans ph;
  random day(ph);
```

gives the results:

Dependent Variable: LOGVAR

Source	DF	Sum of Squares	Mean Square	F Value
Model	17	10.32849194	0.60755835	138.29
Error	108	0.47448146	0.00439335	
Corr. Total	125	10.80297340		

R-Square	Pr > F	Root MSE	LOGVAR Mean	C.V.
0.956079	0.0001	0.066282	-.92821182	-7.140862

Source	DF	Type I SS	Mean Square	F Value	Pr > F
LD	1	8.54219513	8.54219513	1944.35	0.0001
PH	2	1.60099025	0.80049512	182.21	0.0001
D(PH)	6	0.12027099	0.02004517	4.56	0.0004
LD*PH	2	0.01070209	0.00535104	1.22	0.2999
LD*D(PH)	6	0.05433348	0.00905558	2.06	0.0637

Source	DF	Type III SS	Mean Square	F Value	Pr > F
LD	1	8.54219513	8.54219513	1944.35	0.0001
PH	2	0.10533370	0.05266685	11.99	0.0001
D(PH)	6	0.05593527	0.00932254	2.12	0.0565
LD*PH	2	0.01070209	0.00535104	1.22	0.2999
LD*D(PH)	6	0.05433348	0.00905558	2.06	0.0637

where LD=LOGDOSE and D(PH)=DAY(PH).

Least Squares Means

PH	LSMEAN
1	-1.04568416
2	-0.96280088
3	-0.77615044

It follows that the amount of variability explained by pH and dose are orders of magnitude greater than the remaining effects, inclusive the day-to-day variability. Thus the conclusive model will only include the pH and dose effects. The least square means for the three pH levels show the expected ranking.

2.6.6 Conclusion

We have obtained a single feature from digital image analysis to describe the effect of cellulase enzymatic treatment of textiles. This feature is also fast to compute and seems to be robust. Other features measuring the variation in the textile that is coarser than the regular textile pattern can possibly describe the same textile properties, but the coarse-scale normalized variance seems to be the feature that has the overall best performance of the features considered. The feature may also be useful in e.g. carpet wear assessment.

2.7 GLCM feature performance

The performance of 15 GLCM features is tested in CART classification of 15 Brodatz textures. We thereby investigate how much textural information is contained in the simultaneous distribution of (horizontal) neighbor pixels. The cooccurrence matrices are computed on the raw textures, on the textures after a histogram equalization, and on the textures after a Gaussian histogram match.

2.7.1 Image material

15 Brodatz textures were selected on the basis that they should have a fine-grained and homogeneous texture. A part of each of these textures are shown raw in figure 2.8, after a histogram equalization in figure 2.9, and after a Gaussian histogram match in figure 2.10. The names of the selected textures are shown in figure 2.11. The textures D16, D21, D53, D77 and D84 will subsequently be called deterministic due to their relatively strict ordering. The rest will be called stochastic. This grouping will be helpful in the interpretation of the classification results.

The textures were scanned from the paper with an 8-bit, 300 dpi scanner. The output from the scanner is a 2400x1800 image, which is then reduced by two steps in a Gaussian pyramid (Burt, 1981). The approximately Gaussian operator is a separable, symmetric filter with values

$$0.05, 0.25, 0.40, 0.25, 0.05$$

The result is a 600x450 floating point image, where almost no pixels have identical values. Three byte versions of each image is now generated.

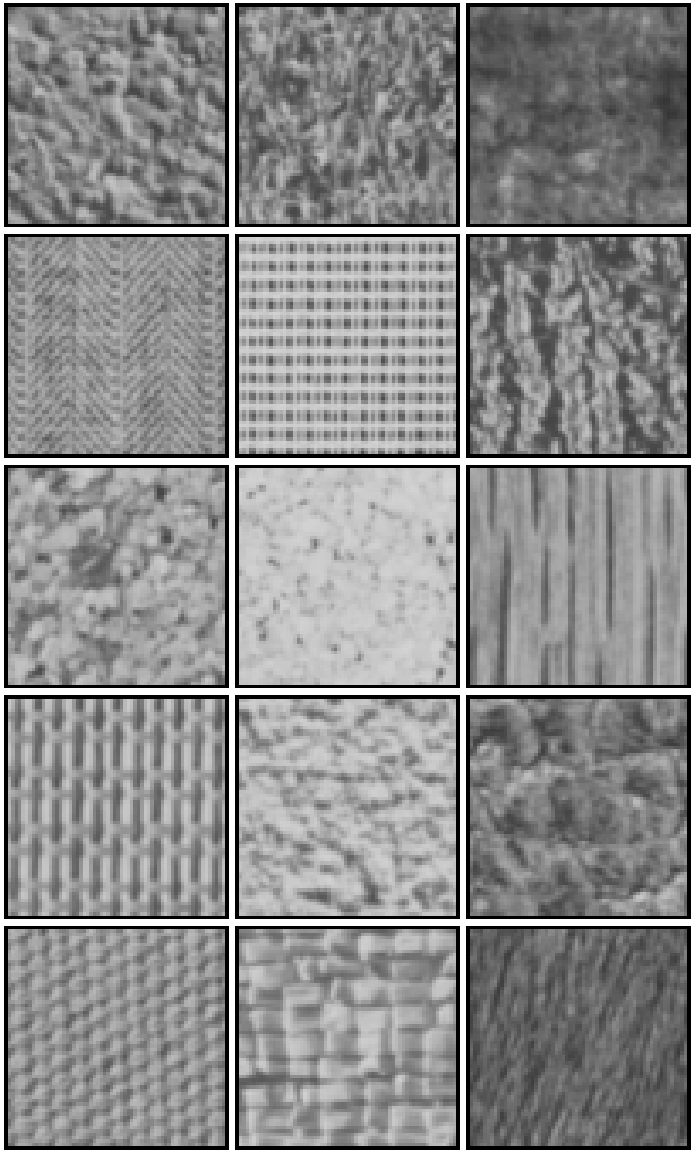

Figure 2.8. 15 Brodatz textures (no histogram match).

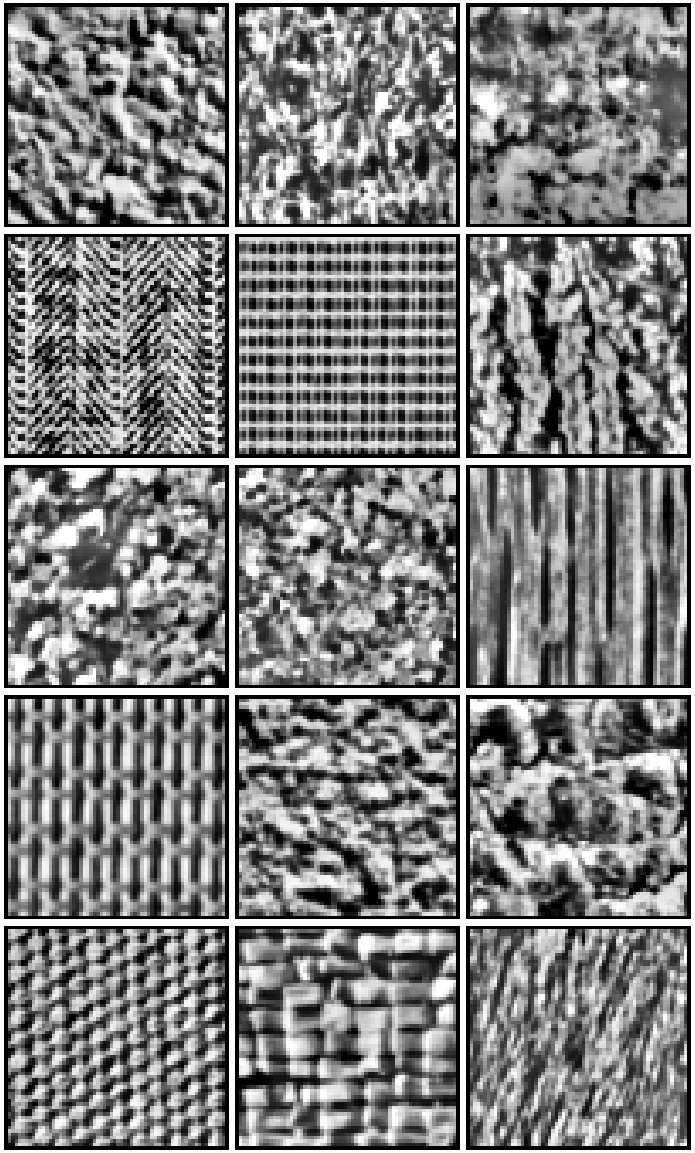

Figure 2.9. 15 Brodatz textures after a histogram equalization.

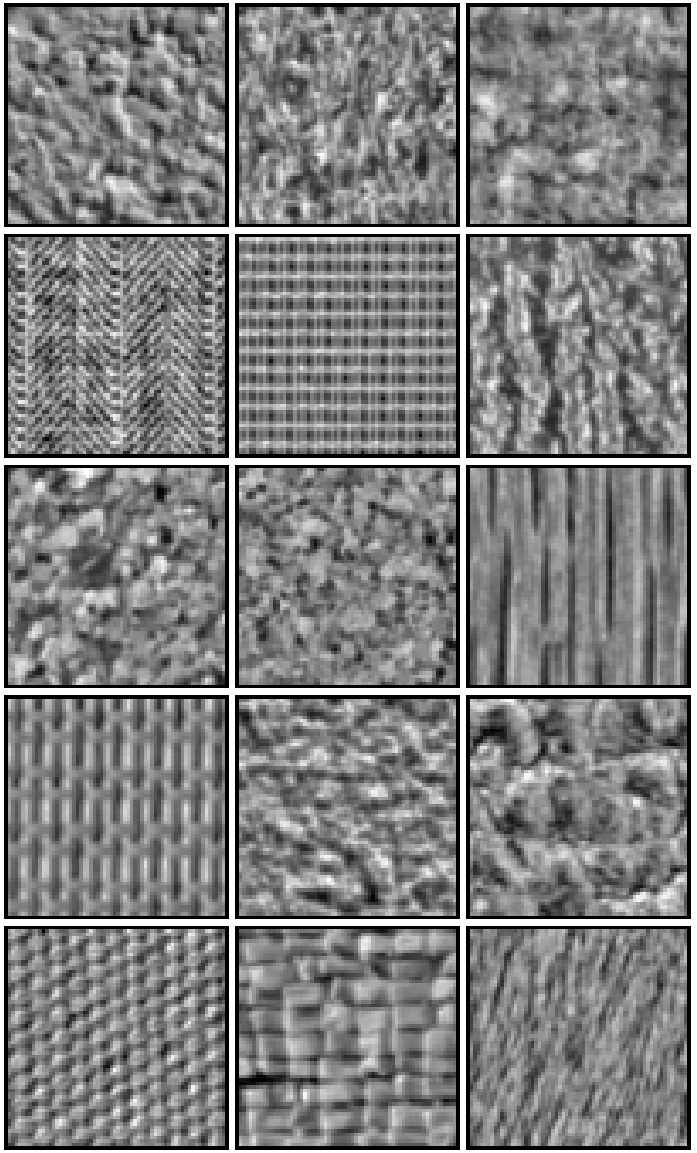

Figure 2.10. 15 Brodatz textures after a Gaussian histogram match.

Pressed cork (D4)	Grass lawn (D9)	Woolen cloth (D19)
Herringbone weave (D16)	French canvas (D21)	Calf leather (D24)
Beach sand (D29)	Pressed cork (D32)	Wood grain (D68)
Oriental straw cloth (D53)	Handmade paper (D57)	Pigskin (D92)
Cotton canvas (D77)	Raffia (D84)	Calf fur (D93)

Figure 2.11. Names of the 15 Brodatz textures in figure 2.10.

- The floating point image scaled linearly.
- A histogram equalized version.
- A Gaussian matched version (mean=127.5, sdev=40.0).

The histogram equalization and Gaussian match are performed by sorting all pixels, while the image is in floating point format, and then assign byte values according to the desired histogram. Thus we obtain a perfect histogram match. Histogram equalization has been used frequently (e.g. Haralick et al. (1973) and Laws (1980)) by researchers studying the performance of texture features. The equalization has in these cases been made using a less accurate byte to byte match.

For the selected, fine-grained textures we corrected for background variations by subtracting a 25x25 median filtered version of each texture from itself.

2.7.2 GLCM

We computed the right-neighbor GLCM (**h**=(0,1)) for the three versions of all 15 textures. In figures 2.12, 2.13 and 2.14 are shown plots of the cooccurrence matrices for respectively the raw versions, the histogram equalized versions and the Gaussian matched versions.

Each image was partitioned in 108 disjoint 50x50 subimages. From the right-neighbor GLCM of these subimages we computed the following GLCM features:

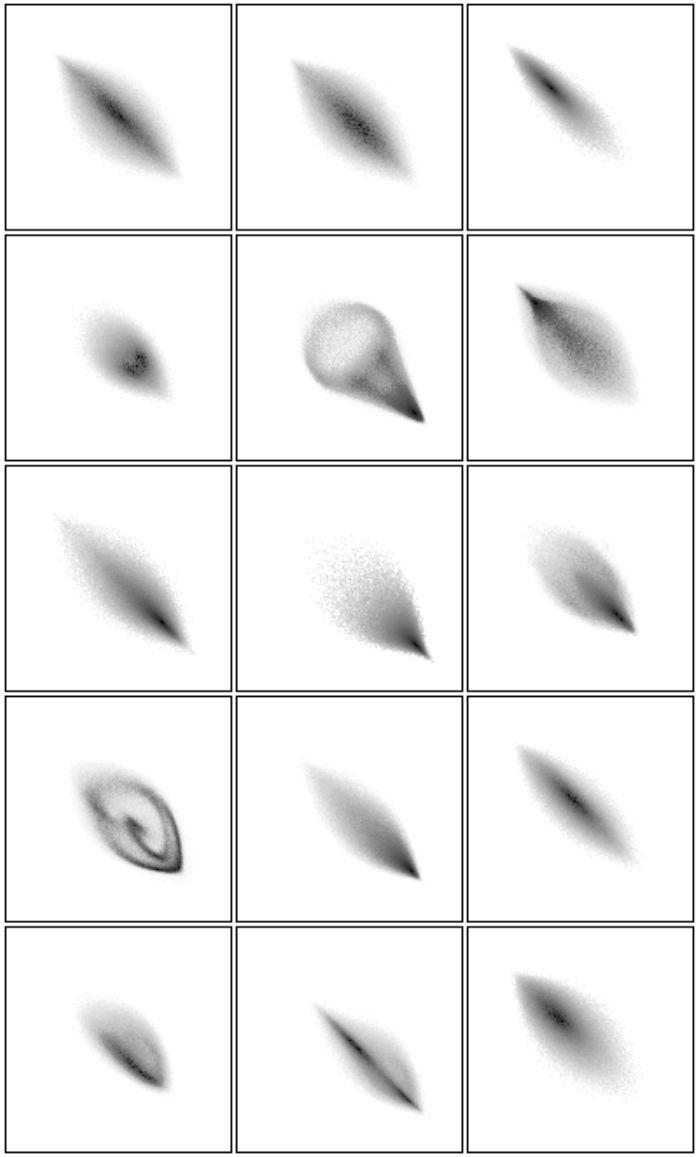

Figure 2.12. Cooccurrence matrices of raw textures.

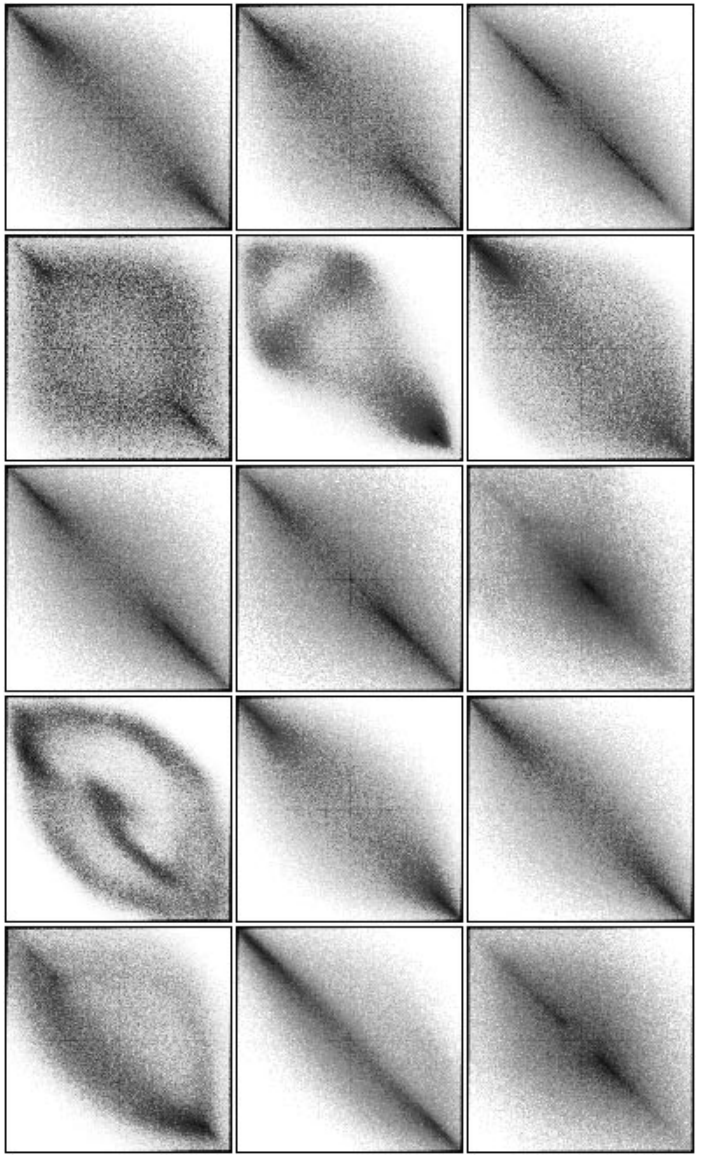

Figure 2.13. Cooccurrence matrices of histogram equalized textures.

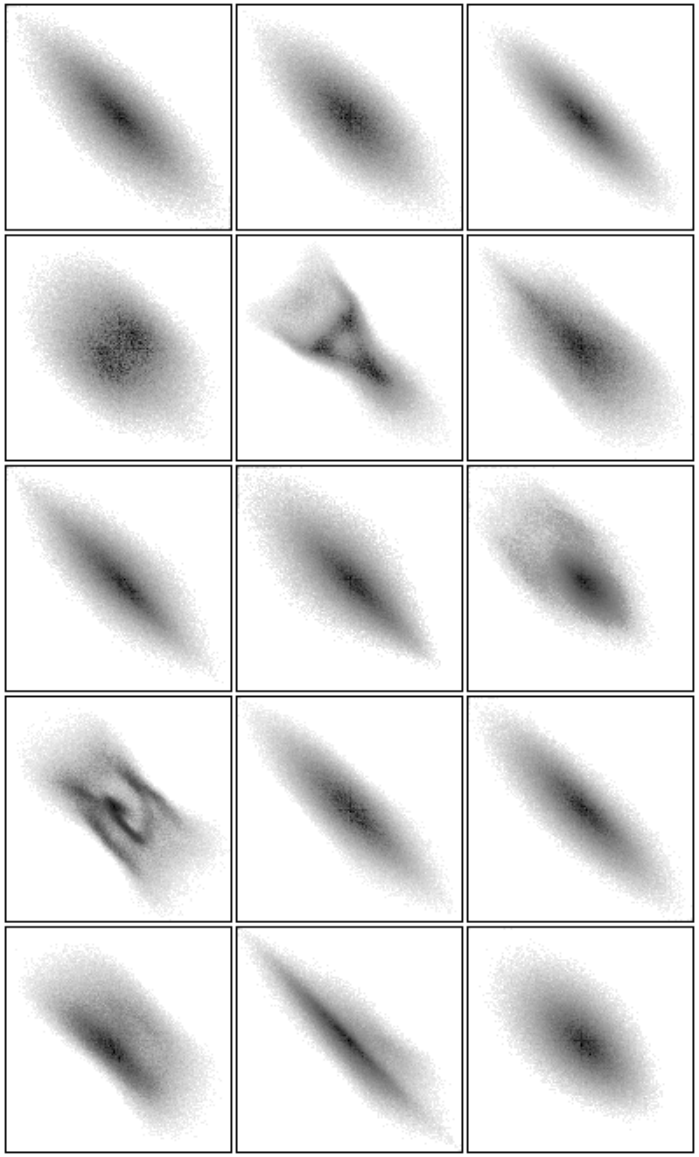

Figure 2.14. Cooccurrence matrices of Gaussian matched textures.

1. Energy (Enrg)
2. Entropy (Entr)
3. Maximum probability (Maxp)
4. Correlation (Corr)
5. Diagonal moment (Diag)
6. Kappa (Kapp)
7. Difference energy (Derg)
8. Difference entropy (Dent)
9. Inertia (Iner)
10. Inverse difference moment (IDM)
11. Sum energy (Serg)
12. Sum entropy (Sent)
13. Sum variance (Svar)
14. Cluster shade (Shad)
15. Cluster prominence (Prom)

2.7.3 CART classification

Classification and regression trees is a nonparametric alternative to classical discriminant analysis. A binary decision tree is constructed and a classification is made by running down the tree and choose the class corresponding to

the terminal node. The CART program from California Statistical Software, Inc. was used. The reader is referred to Breiman, Friedman, Olshen, & Stone (1984) and the CART documentation for detailed information about CART.

Only splits based on single features were allowed. 10-fold cross-validation was used for estimating the probability of correct classification.

We made a CART classification on seven subsets of the 15 textures in all three versions. The seven subsets are:

1. The five textures in the left column.
2. The five textures in the middle column.
3. The five textures in the right column.
4. The five deterministic textures: D16, D21, D53, D77 and D84.
5. The five stochastic textures: D4, D9, D29, D32 and D57.
6. The ten textures in the left and middle columns.
7. All 15 textures.

Linearly scaled versions

In figure 2.15 we see the classification tree suggested by CART for set 2. A texture is classified by starting at the top node and then run down the tree until a terminal node is reached. Every terminal node is associated with a texture class, and this class is assigned to the texture that we wish to

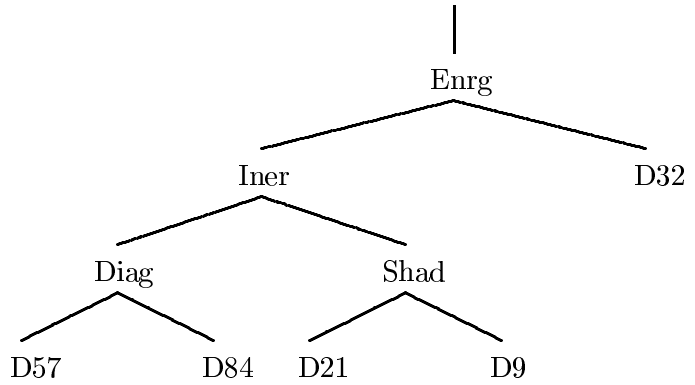

Figure 2.15. Classification tree for set 2 with no histogram match.

classify. At every nonterminal node a decision is made based on the value of one feature. If the value of the feature is lower than the split value for that node we go left in the tree, otherwise we go right. This classification is a partitioning of feature space into boxes. The cross-validation estimate was one misclassified texture out of 540. CART also showed that many alternative trees would have a similar performance. Figure 2.16 shows a scatter plot of the inertia versus the diagonal moment for the textures in set 2. We see that the texture classes are easily discriminated.

When no histogram match is performed the first-order statistics will influence the cooccurrence features. As the first-order statistics of the Brodatz textures differ significantly, the set of 15 GLCM features will be able to discriminate between any subset of the 15 textures (actually even any subset of all the 112 Brodatz textures) with close to 0% error rate. Hence we shall concentrate on the histogram matched versions.

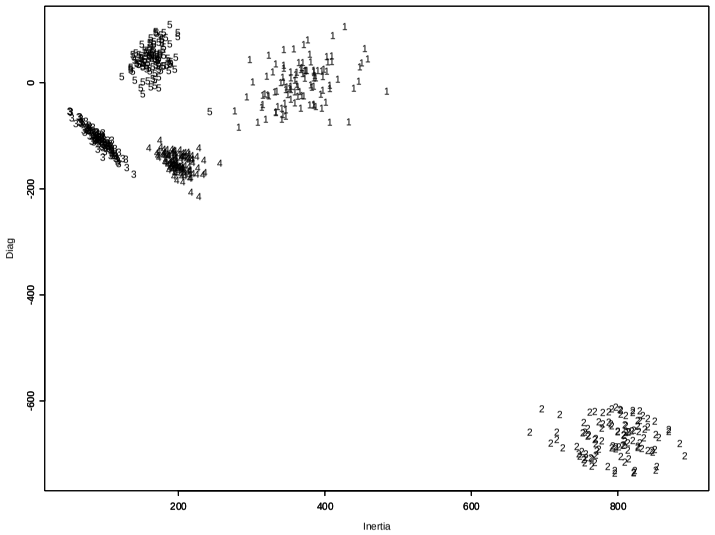

Figure 2.16. Scatter plot of the diagonal moment versus the inertia for the textures in set 2. 1=D9, 2=D21, 3=D32, 4=D57, 5=D84.

Histogram equalized versions

A summary of the classification results for the histogram equalized textures is listed in the following table.

Set no.	Classes	Terminal nodes	% correctly classified	Most important feature
1	5	10	95.9	Iner
2	5	10	96.5	Derg
3	5	6	80.9	Svar
4	5	6	98.5	Corr
5	5	13	81.3	Corr
6	10	27	89.3	Corr
7	15	54	74.3	Derg

The results show that the sets with several deterministic textures has higher percentage of correctly classified textures. i.e. the deterministic textures in this study are relatively easy to discriminate. The correlation feature and uniformity features based on energy and entropy are important for the classification.

We shall now study the classification results of set 2 in more detail. The classification tree is shown in figure 2.17. The first split is based on the correlation feature, and it discriminates the textures D57 and D84 from the other three. This is a good split (high discriminatory power) and so are the two splits on the second level. However on the right branch of the split based on the diagonal moment we see a relatively complex subtree trying to discriminate between the textures D9 and D32. The features used for this purpose are difference energy, energy and entropy. Figure 2.18 shows a

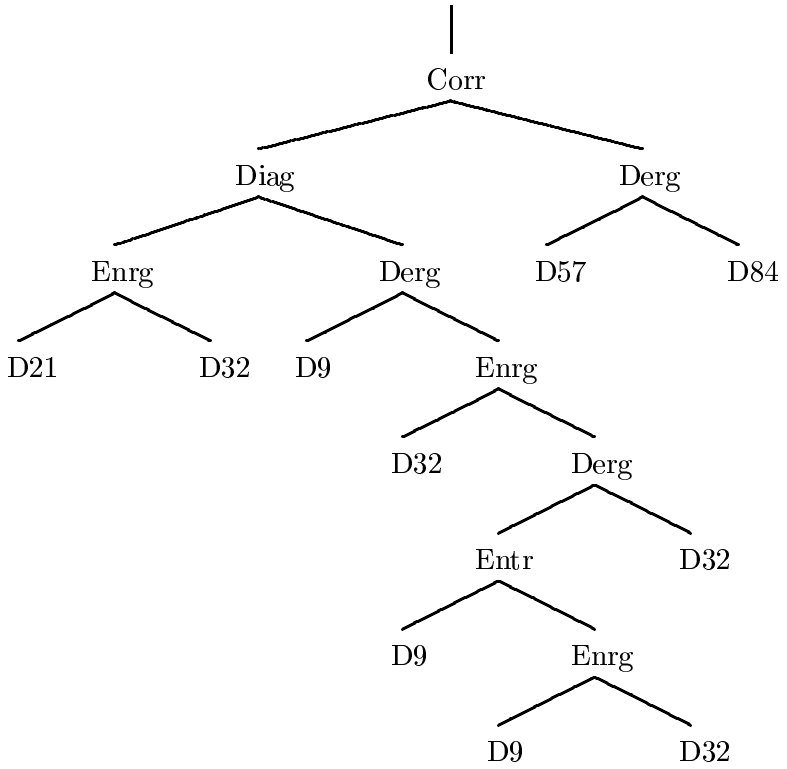

Figure 2.17. Classification tree for set 2 after histogram equalization.

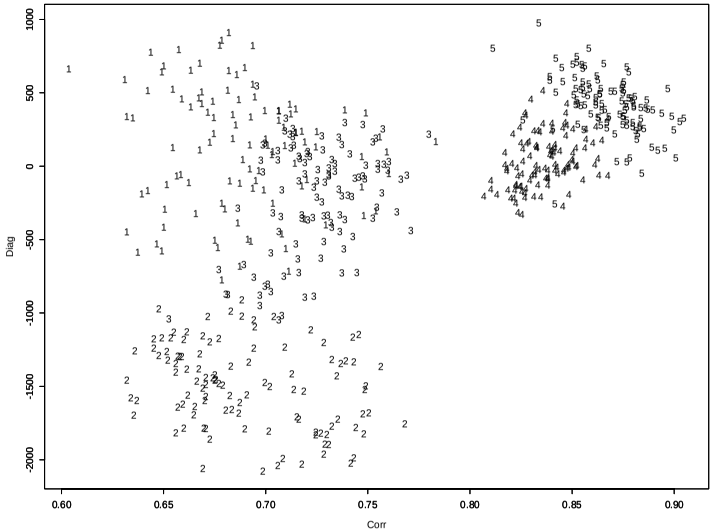

Figure 2.18. Scatter plot of the diagonal moment versus the correlation for the textures in set 2 after histogram equalization. 1=D9, 2=D21, 3=D32, 4=D57, 5=D84.

scatter plot of the diagonal moment versus the correlation. We see that the major discriminatory deficiency in these two features is the mixture of the classes D9 and D32. Figure 2.19 shows a scatter plot of the energy versus the difference energy. It is obvious that there is no easy way out of the discriminatory problem.

Gaussian matched versions

A summary of the classification results for the Gaussian matched textures is listed in the following table.

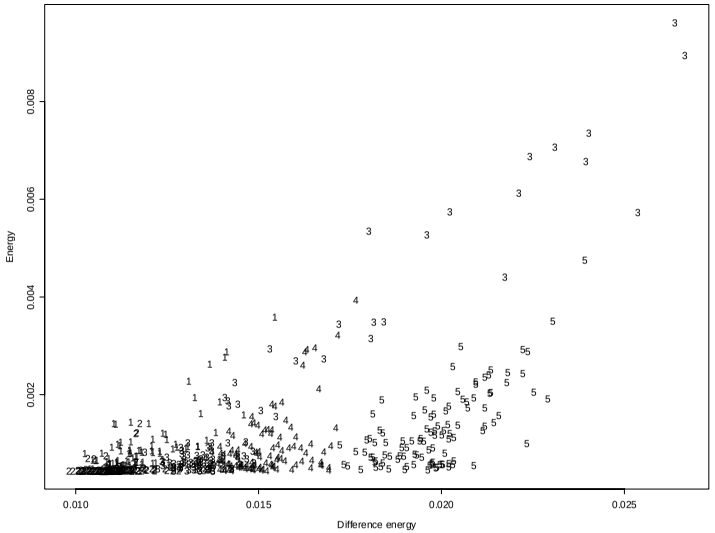

Figure 2.19. Scatter plot of the energy versus the difference energy for the textures in set 2 after histogram equalization. 1=D9, 2=D21, 3=D32, 4=D57, 5=D84.

Set no.	Classes	Terminal nodes	% correctly classified	Most important feature
1	5	7	93.5	Corr
2	5	6	97.6	Diag
3	5	8	85.9	Diag
4	5	5	97.2	Corr
5	5	6	84.3	Corr
6	10	18	89.7	Corr
7	15	40	80.9	Corr

Again we see that the deterministic textures are relatively easy to discriminate. All sets except the sets with a majority of deterministic textures (set 1 and set 4) were classified more correctly with these features than with the features based on histogram equalization. Generally the diagonal moment was an important feature, and for two sets even the most important. It can also be seen that in general the trees has fewer terminal nodes than trees based on the histogram equalization, thus we get simpler trees. The energy and the entropy features were found to be highly correlated for all 15 textures as were the difference entropy and the inertia.

The classification tree for set 2 is shown in figure 2.20. The tree is simpler than the tree based on the histogram equalization. Only the three features correlation, diagonal moment and inverse difference moment are used. Figure 2.21 shows a scatter plot of the diagonal moment versus the correlation. There is hardly any confusion between the classes D9 and D32.

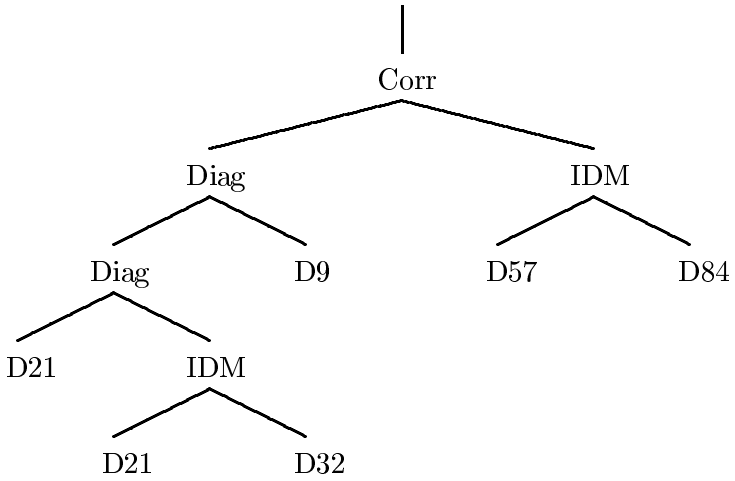

Figure 2.20. Classification tree for set 2 after Gaussian histogram match.

2.7.4 Classification summary

The results of the classifications are summarized as follows:

- It is easy to discriminate the Brodatz textures if no histogram match is performed.
- Features based on a Gaussian match performed better than features based on histogram equalization for the stochastic textures.
- Features based on histogram equalization performed a little better than features based on a Gaussian match for the deterministic textures.

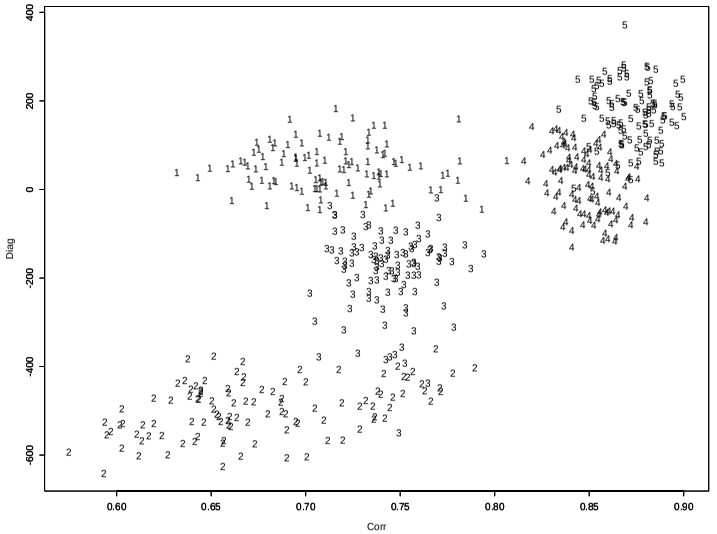

Figure 2.21. Scatter plot of the diagonal moment versus the correlation for the textures in set 2 after Gaussian histogram match. 1=D9, 2=D21, 3=D32, 4=D57, 5=D84.

- The deterministic textures were easier to discriminate than the stochastic textures.
- Features based on histogram equalization generally produce trees with more nodes than features based on a Gaussian match.
- Generally correlation was the most important feature.
- The diagonal moment was a very important feature. Many splits were based on the diagonal moment.
- The uniformity features energy, entropy, difference energy, difference entropy, sum energy and sum entropy seems to be more important for histogram equalized textures.
- The maximum probability feature was generally unimportant.
- The energy and the entropy features were highly correlated as were the difference entropy and the inertia.

2.7.5 Conclusion

The performance of 15 right-neighbor GLCM features in CART classification of 15 Brodatz textures has been investigated.

This study has shown that histogram matching of textures has a significant effect on the discriminatory performance of GLCM features computed from the textures. Especially it seems that histogram equalization is too crude for stochastic textures. For such textures a Gaussian match will give better performance and a simpler and more interpretable classifier. The Brodatz textures are easily discriminated if no histogram match is made.

The diagonal moment is an important feature. As this feature can not be computed from the gray level difference histogram (GLDH) and the gray level sum histogram (GLSH), there is a loss of relevant information when replacing the GLCM with these two histograms.

Generalization of the conclusions of this study should be done with great caution. The selection of 15 textures that we used represent an insignificant fraction of real-world textures, and only the horizontal neighbor relation has been investigated.

Chapter 3

Markov random fields

This chapter deals with parametric description of texture based on a class of models called *Markov random fields*. The theory of Markov Random fields is reviewed together with the theory of the associated *Gibbs random fields*. The theory of Gibbs random fields were founded in statistical physics (Ising, 1925) and some relevant results from this area is presented in a new statistical setting. A variety of Markov random fields is reviewed with an emphasis on discrete models. Further we introduce a set of morphological Markov random fields, that extends the standard set of models by using the operators of mathematical morphology (Serra, 1982).

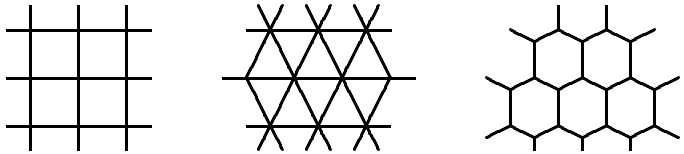

Figure 3.1. *Regular 2D tessellations. Rectangular, triangular and hexagonal.*

3.1 Random fields

One of the main tasks in statistical image processing is to construct stochastic models for observed images and especially for textures. The pixel values $\{x_i, i = 0,1,..,n-1\}$ are represented as realizations of random variables $\{X_i, i = 0,1,..,n-1\}$, and the probability measure representing the joint distribution of all pixel values on an image grid is called a *random field*. $P(\mathbf{x})$ is the probability of a particular image or configuration $\mathbf{x} \in \Omega$, where Ω is the set of all possible configurations on the given grid.

3.1.1 2D grids

There exists three ways of partitioning the two-dimensional plane in disjunct, regular polygons of equal size. Such a partitioning is called a regular tessellation. The three regular tessellations are the regular square tessellation, the regular triangular tessellation and the regular hexagonal tessellation as shown in figure 3.1.

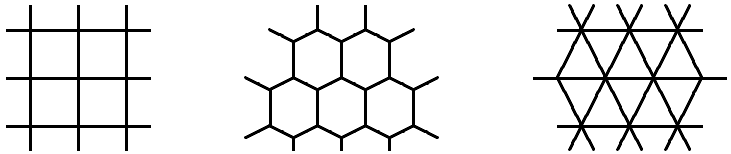

Figure 3.2. *2D pixel grids. Rectangular, honeycomb and hexagonal.*

Let the polygons of a tessellation correspond to pixels, then the graph corresponding to the pixel grid will be dual to the graph of polygon borders, i.e.

- A square tessellation correspond to a square pixel grid
- A triangular tessellation correspond to a honeycomb pixel grid
- A hexagonal tessellation correspond to a triangular (hexagonal) pixel grid

The honeycomb grid is used in statistical mechanics but very rarely (if at all) used in image analysis. Since the neighborhood of a pixel in a triangular grid is hexagonal, and the pixels are hexagonal, this grid is often called the hexagonal grid, even though this term fits just as well for the honeycomb grid. Here we will follow the common practice in image analysis, i.e. subsequently a hexagonal grid has hexagonal pixels. The hexagonal grid is quite popular in mathematical morphology (Serra, 1982) due to the attractive neighborhood structure. The square grid is used in the vast majority of situations, and where nothing else is mentioned this will be synonymous to grid. The grids corresponding to the tessellations of figure 3.1 are shown in figure 3.2. Pixels are located at the line intersections.

3.2 Gibbs random fields

3.2.1 Historical perspective

In 1877 Boltzmann investigated the distribution of energy states in molecules of an ideal gas. According to the *Boltzmann distribution* the probability of a molecule being in a state with energy ε is:

$$P(\varepsilon) = \frac{1}{z} e^{-\frac{1}{kT}\varepsilon}$$

where z is a normalization constant, that makes the probabilities sum to one. T is the *absolute temperature*, and k, *Boltzmann's constant*, is a constant of nature, that relates temperature to energy. In all subsequent formulas the temperature will be assumed measured in energy units, hence kT will be replaced by T.

Gibbs used a similar distribution in 1901 to express the probability of a whole system with many degrees of freedom being in a state with a certain energy. Let $\mathbf{x}$ denote a state in state space Ω and $U : \Omega \mapsto \mathcal{R}$ be the *energy function*. Then

$$P(\mathbf{X} = \mathbf{x}) = \frac{1}{Z} e^{-\frac{1}{T}U(\mathbf{x})} \tag{3.1}$$

where

$$Z = \sum_{\mathbf{x}\in\Omega} e^{-\frac{1}{T}U(\mathbf{x})}.$$

Z is called the *partition function*. T controls the degree of peaking in the probability density function. As $T \to \infty$ the distribution will tend to a uniform distribution among all possible states. As $T \to 0$ the distribution will tend to a uniform distribution among the minimum energy states. The distribution 3.1 is called the *Gibbs distribution* or *canonical distribution*. Subsequently the former term will be used exclusively.

Ising (1925) used the Gibbs distribution to describe the behavior of ferromagnetic materials. Any site or pixel in such a material is thought of as a small dipole, which can be in state "spin up" or "spin down" corresponding to values 1 and -1.

The *Ising model* on a square grid is defined through the energy function

$$U(\mathbf{x}) = -J \sum_{i \sim j} x_i x_j - mH \sum_i x_i$$

where $i \sim j$ means that pixel i and pixel j are either horizontal or vertical nearest neighbors. J is a property of the material that determines the interaction between neighboring spins. If $J > 0$ neighboring spins tend to be equal. If $J < 0$ neighboring spins tend to be opposite. $J = 0$ means no interaction. The constant $m > 0$ is a property of the material that determines the sensitivity of the spins to an external magnetic field of intensity H. $H > 0$ will favor a spin up, whereas $H < 0$ will favor a spin down. The Ising model has been successful in explaining ferromagnetic phenomena, but has also founded an interest in the more general Gibbs random fields.

Brush (1967) reviews the history of the Ising model.

3.2.2 General properties

Gibbs random fields are random fields defined through equation 3.1. This means that for every energy function on Ω there exists a corresponding Gibbs random field. Not all of these Gibbs random fields are useful for our purposes and in the next section we shall limit our attention to a very interesting subclass.

The Gibbs measure has an interesting property with respect to *entropy*. The entropy S is frequently used as a uniformity measure of a random field P, and is defined as

$$S(P) = -\sum_{\mathbf{x} \in \Omega} P(\mathbf{x}) \log P(\mathbf{x}).$$

Of all probability measures defined through an energy function the Gibbs measure (3.1) is the measure which maximizes entropy among all measures with the same expected energy (Jaynes, 1957).

3.3 Markov random fields

Hassner & Sklansky (1980) introduced Markov random fields to image analysis and through the last decade Markov random fields have been used extensively as representations of visual phenomena. In this thesis there is put a strong emphasis on Markov random fields with discrete pixel values i.e. discrete Markov random fields, but most of the results are easily extended to continuous Markov random fields. For more thorough expositions on Markov random fields the reader is referred to Geman (1990),Dubes & Jain (1989), and Ripley (1988).

In the rest of this section we shall restate some definitions regarding Markov random fields and a theorem that shows an equivalence between Markov random fields and Gibbs random fields.

Definition 1 . *Let $S = \{s_0, s_1, ..., s_{n-1}\}$ be a set of sites. A* **neighborhood system** *$\mathcal{N} = \{N_s, s \in S\}$ is a collection of subsets of S for which*

1. $s \notin N_s$
2. $r \in N_s \Leftrightarrow s \in N_r$

N_s *are the* **neighbors** *of* s.

When sites i and j are neighbors we write $i \sim j$. The set of all possible configurations on S is called Ω.

Definition 2 . *A* **clique** C *is a subset of* S *for which every pair of sites are neighbors.*

Single pixels are also considered cliques. The set of all cliques on a grid is called $\mathcal{C}$.

Definition 3 . *A random field* $\mathbf{X}$ *is a* **Markov random field (MRF)** *with respect to the neighborhood system* $\mathcal{N} = \{N_s, s \in S\}$ *iff*

1. $P(\mathbf{X} = \mathbf{x}) > 0$ *for all* $\mathbf{x} \in \Omega$
2. $P(X_s = x_s | X_r = x_r, r \neq s) = P(X_s = x_s | X_r = x_r, r \in N_s)$ *for all* $s \in S$ *and* $\mathbf{x} \in \Omega$

The structure of the neighborhood system determines the *order* of the MRF. For a first order MRF the neighborhood of a pixel consists of its four nearest neighbors. In a second order MRF the neighborhood consists of the eight nearest neighbors. The clique structures are illustrated in figure 3.3 and

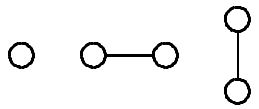

Figure 3.3. *Cliques for a first-order neighborhood.*

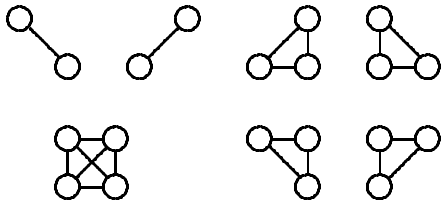

Figure 3.4. *Additional cliques for a second-order neighborhood.*

5	4	3	4	5
4	2	1	2	4
3	1	.	1	3
4	2	1	2	4
5	4	3	4	5

Figure 3.5. Order coding of neighborhood structure. The n-order neighborhood of the center pixel (.) contains the pixels with numbers less than or equal to n.

figure 3.4 for a first-order MRF and a second-order MRF. The order coding of the neighborhood up to order five is shown in figure 3.5.

Definition 4 . **X** *is a* **Gibbs random field (GRF)** *with respect to the neighborhood system* $\mathcal{N} = \{N_s, s \in S\}$ *iff*

$$P(\mathbf{X} = \mathbf{x}) = \frac{1}{Z} \exp(-U(\mathbf{x})/T)$$

where Z *is a normalizing constant called the* **partition function**, T *is a control parameter called* **temperature** *and* U *is the* **energy function** *of the form*

$$U(\mathbf{x}) = \sum_{C \in \mathcal{C}} V_C(\mathbf{x})$$

where V_C *is called a* **potential** *and is a function depending only on* $x_s, s \in C$,

Theorem 1 . *(Hammersley-Clifford). A random field* **X** *is a Gibbs random field with respect to the neighborhood system* $\mathcal{N}$ *iff* **X** *is a Markov random field with respect to* $\mathcal{N}$.

A simple proof may be found in Geman (1990). Using this equivalence we have both a local and a global description of the distribution. In the present context we use the term Markov random field to emphasize the Markov property.

3.4 Binary Markov random fields

3.4.1 Ising model revisited

The best known and most investigated Markov random field is the Ising model. This model has been studied in statistical physics since its introduction in Ising (1925), whereas statisticians joined the efforts in the 1960's. We shall give a thorough description of the Ising model using statistical terminology. Thus with the notation introduced in the previous section we will talk about the first-order binary Markov random field. The reader is referred to Kinderman & Snell (1980) for background material on this issue.

In our notation every site can take the values 0 or 1. The neighborhood of a pixel is the four nearest neighbors. The corresponding three cliques are single pixels, horizontal neighbors and vertical neighbors. Single pixels with value one have the potential $-\alpha$. Horizontal neighbor cliques have the potential $-\beta_1$ if both pixels are one. The corresponding vertical neighbor clique potential is $-\beta_2$. If any pixel in a clique is 0 the clique potential is 0.

This gives us the energy function

$$U(\mathbf{x}) = -\alpha \sum_i x_i - \beta_1 \sum_{i \leftrightarrow j} x_i x_j - \beta_2 \sum_{i \updownarrow j} x_i x_j \tag{3.2}$$

and the joint distribution

$$P(\mathbf{X} = \mathbf{x}) = \frac{1}{Z(\alpha, \beta_1, \beta_2)} \exp(\alpha \sum_i x_i + \beta_1 \sum_{i \leftrightarrow j} x_i x_j + \beta_2 \sum_{i \updownarrow j} x_i x_j) \tag{3.3}$$

where $i \leftrightarrow j$ means that i and j are horizontal neighbors, and $i \updownarrow j$ means that i and j are vertical neighbors. If $\beta_1 = \beta_2$ the configurations will show

no directionality and we call this an *isotropic model*. The more general formulation in (3.3) represents the *anisotropic model*. The joint distribution for the isotropic model is

$$P(\mathbf{X}=\mathbf{x}) = \frac{1}{Z(\alpha,\beta)}\exp(\alpha\sum_i x_i + \beta\sum_{i\sim j} x_i x_j)$$

where $i \sim j$ means that i and j are neighbors.

The expected mean and variance can be expressed as

$$E(\sum_i X_i) = \sum_\Omega [\sum_i x_i]P(\mathbf{x}) = \frac{1}{Z}\frac{\partial}{\partial\alpha}Z = \frac{\partial}{\partial\alpha}\log Z$$

$$V(\sum_i X_i) = \sum_\Omega [\sum_i x_i]^2 P(\mathbf{x}) - \frac{1}{Z^2}(\frac{\partial}{\partial\alpha}Z)^2$$

$$= \frac{1}{Z}\frac{\partial^2}{\partial\alpha^2}Z - \frac{1}{Z^2}(\frac{\partial}{\partial\alpha}Z)^2 = \frac{\partial^2}{\partial\alpha^2}\log Z.$$

This result is valid for both the isotropic and anisotropic models.

In the horizontal direction we get

$$E(\sum_{i\leftrightarrow j} X_i X_j) = \sum_\Omega [\sum_{i\leftrightarrow j} x_i x_j]P(\mathbf{x}) = \frac{1}{Z}\frac{\partial}{\partial\beta_1}Z = \frac{\partial}{\partial\beta_1}\log Z$$

$$V(\sum_{i\leftrightarrow j} X_i X_j) = \sum_\Omega [\sum_{i\leftrightarrow j} x_i x_j]^2 P(\mathbf{x}) - \frac{1}{Z^2}(\frac{\partial}{\partial\beta_1}Z)^2$$

$$= \frac{1}{Z}\frac{\partial^2}{\partial\beta_1^2}Z - \frac{1}{Z^2}(\frac{\partial}{\partial\beta_1}Z)^2 = \frac{\partial^2}{\partial\beta_1^2}\log Z.$$

For the vertical direction and for the isotropic case the results are analogous.

As it can be seen from the equations above the partition function is a main key in understanding and describing the behaviour of this model.

Many attempts have been made to make evaluation of the partition function possible. The only exact result was found by Onsager (1944) for the zero-field Ising model in the large grid limit. Zero-field means that the marginal probability of 0-pixels and 1-pixels are equal, i.e. $\alpha = -\beta_1 + \beta_2$. Let N be the number of pixels in the grid. Onsager found that in the limit $N \to \infty$ we can write $\frac{1}{N}\log Z$ as

$$\log 2 - \frac{\beta_1+\beta_2}{2}$$
$$+\frac{1}{2\pi^2}\int_0^\pi\int_0^\pi \log(\cosh\tfrac{\beta_1}{2}\cosh\tfrac{\beta_2}{2} - \sinh\tfrac{\beta_1}{2}\cos\omega_1 - \sinh\tfrac{\beta_2}{2}\cos\omega_2)d\omega_1 d\omega_2.$$

Using this expression we can find the correlation between horizontal neighbors in the limit $N \to \infty$ as

$$\Gamma_1(\beta_1,\beta_2) = E(\frac{4}{N}\sum_{i\leftrightarrow j} X_i X_j - 1)$$

$$= \frac{1}{2\pi^2}\int_0^\pi\int_0^\pi \frac{2\sinh\frac{\beta_1}{2}\cosh\frac{\beta_2}{2} - 2\cosh\frac{\beta_1}{2}\cos\omega_1}{\cosh\frac{\beta_1}{2}\cosh\frac{\beta_2}{2} - \sinh\frac{\beta_1}{2}\cos\omega_1 - \sinh\frac{\beta_2}{2}\cos\omega_2}d\omega_1 d\omega_2. \tag{3.4}$$

An analogous expression is obtained for the vertical neighbor correlation, $\Gamma_2(\beta_1,\beta_2)$. In the isotropic case we get the nearest neighbor correlation in the limit $N \to \infty$ as

$$\Gamma(\beta) = E(\frac{2}{N}\sum_{i\sim j} X_i X_j - 1)$$

$$= \frac{1}{2\pi^2}\int_0^\pi\int_0^\pi \frac{2\sinh\frac{\beta}{2}\cosh\frac{\beta}{2} - \cosh\frac{\beta}{2}(\cos\omega_1+\cos\omega_2)}{\cosh^2\frac{\beta}{2} - \sinh\frac{\beta}{2}(\cos\omega_1+\cos\omega_2)}d\omega_1 d\omega_2. \tag{3.5}$$

The integrals can be computed by numerical integration, e.g. using Gaussian quadratures (Press, Flannery, Teukolsky, & Vetterling, 1988). Figures 3.6 and 3.7 show plots of $\Gamma_1(\beta_1,\beta_2)$ and $\Gamma(\beta)$.

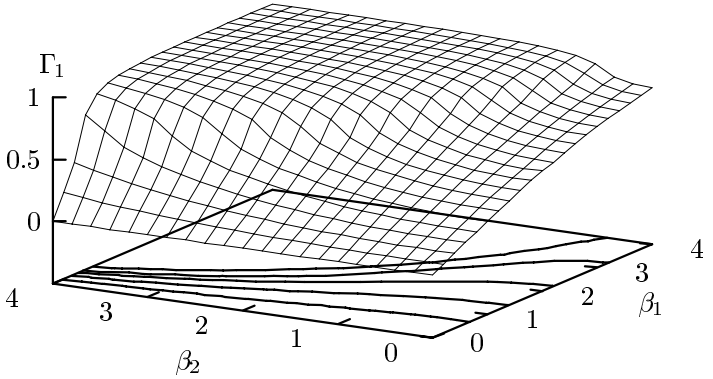

Figure 3.6. Nearest horizontal neighbor correlation versus β_1 and β_2 for the anisotropic model in the large grid limit. The lines in the (β_1,β_2) plane are isolines for the correlation.

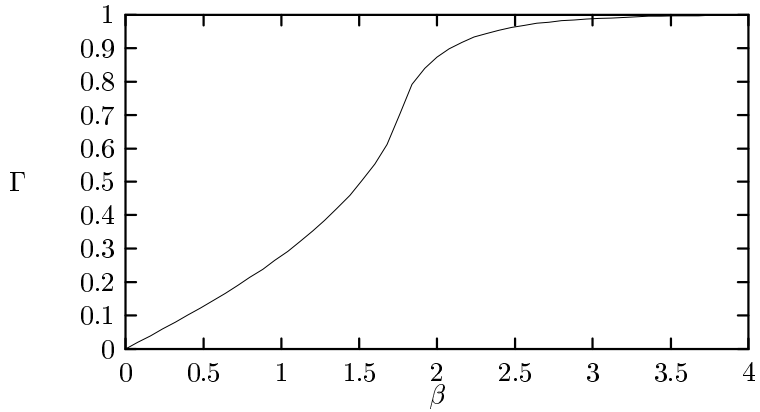

Figure 3.7. Nearest neighbor correlation versus β for the isotropic model in the large grid limit (Pickard,1987).

The local properties of the model is determined by the conditional probabilities

$$P(X_i = x_i | x_w, x_e, x_n, x_s) = \frac{\exp(x_i[\alpha + \beta_1(x_w + x_e) + \beta_2(x_n + x_s)])}{1 + \exp(\alpha + \beta_1(x_w + x_e) + \beta_2(x_n + x_s))}$$

where x_n,x_s,x_w and x_e are the north, south west and east neighbors of x_i.

The parameters are easily interpreted in that α controls the number of 1-pixels, β_1 controls the number of horizontal 1-1-neighbors and β_2 controls the number of vertical 1-1-neighbors.

Phase transitions

A phase transition (Kinderman & Snell, 1980; Pickard, 1987) occurs in a MRF when the locally specified interactions are high enough to develop into long-range correlations.

Onsager (1944) showed that the Ising model has a phase transition for

$$\sinh \frac{\beta_1}{2} \sinh \frac{\beta_2}{2} = 1.$$

Figure 3.8 shows the critical parameters in parameter space. For the isotropic model the critical value is $\beta_c = \sinh^{-1} 1 = 1.7627$. We talk about *supercritical* parameters if $\sinh \frac{\beta_1}{2} \sinh \frac{\beta_2}{2} > 1$ and *subcritical* parameters if $\sinh \frac{\beta_1}{2} \sinh \frac{\beta_2}{2} < 1$. From the figure we see that 1D Ising models do not have a phase transition. If we go to the supercritical limit in each of the four quadrants we get the deterministic patterns shown in figure 3.9. There are two such deterministic patterns in each quadrant, one being the pixelwise negation of the other. For the first quadrant we have a black configuration and a white configuration. In the third quadrant we have checkerboard

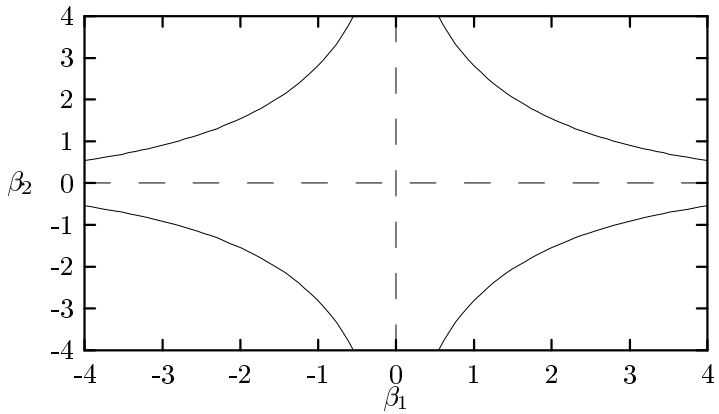

Figure 3.8. Phase transition borders for an anisotropic zero-field Ising model.

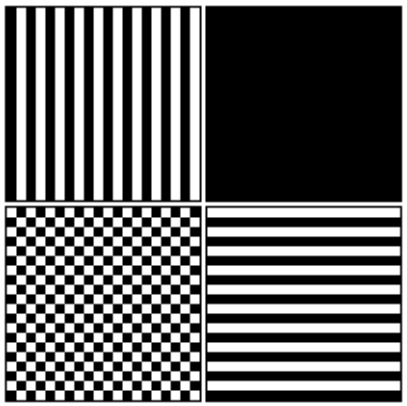

Figure 3.9. Deterministic patterns for each of the four quadrants in the supercritical limit.

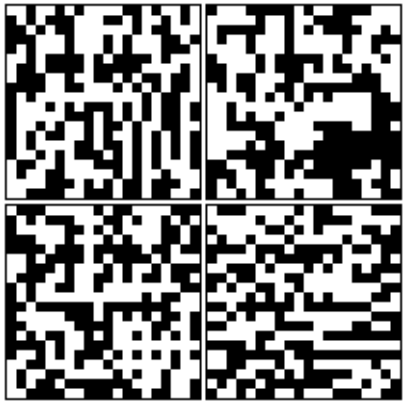

Figure 3.10. Nondeterministic pattern represented in each of the four quadrants. The four patterns can be generated from each other in a very simple way.

and negated checkerboard. We can use this knowledge of the deterministic patterns to understand the relation between nondeterministic patterns in different quadrants. The value of every pixel in a nondeterministic pattern will correspond to the value of the same pixel in one of the two deterministic patterns, i.e. we can partition the image based on deterministic pattern membership. If we then replace pixels belonging to each deterministic pattern with the values of the corresponding deterministic patterns in another quadrant, the result is a transformation of the nondeterministic pattern to the other quadrant. Figure 3.10 shows a nondeterministic pattern represented in all of the four quadrants. The visual symmetry thus obtained elegantly matches the algebraic symmetry of parameter space.

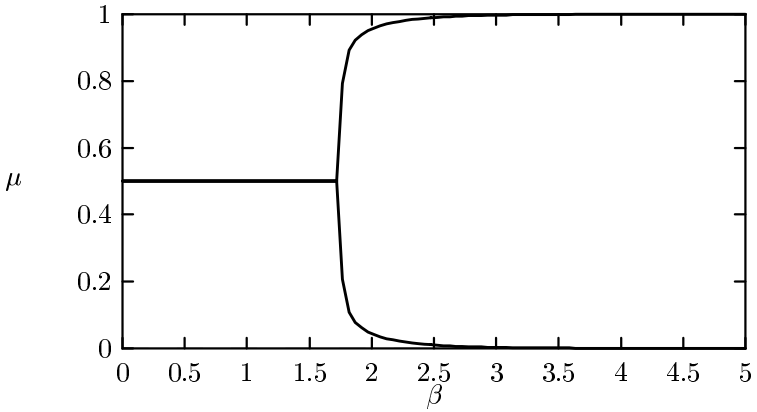

Figure 3.11. The expected fraction of 1-pixels as a function of β for an isotropic zero-field Ising model. The bifurcation point occurs for $\beta = 2\sinh^{-1}(1) = 1.7627$.

An exact expression for the expected fraction of 1-pixels, μ, has been obtained for the zero-field isotropic Ising model in the large grid limit.

$$\mu(\beta) = \begin{cases} \frac{1}{2} + \frac{1}{2}(1 - \frac{1}{(\sinh\frac{\beta}{2})^4})^{\frac{1}{8}} & \text{if } \beta > \beta_c, \text{ white configurations} \\ \frac{1}{2} - \frac{1}{2}(1 - \frac{1}{(\sinh\frac{\beta}{2})^4})^{\frac{1}{8}} & \text{if } \beta > \beta_c, \text{ black configurations} \\ \frac{1}{2} & \text{if } \beta \leq \beta_c \end{cases}$$

This result originated in the work of Onsager (1944) and Yang (1952). In figure 3.11 we see μ plotted versus β. The bifurcation occurring at β_c means zero-field configurations does not have 50% 1-pixels, but 50% of the configurations have almost 100% 1-pixels and the other 50% have almost 100% 0-pixels. The area between the to branches for supercritical β represents configurations with very low probability for all values of α. In figure 3.12 is

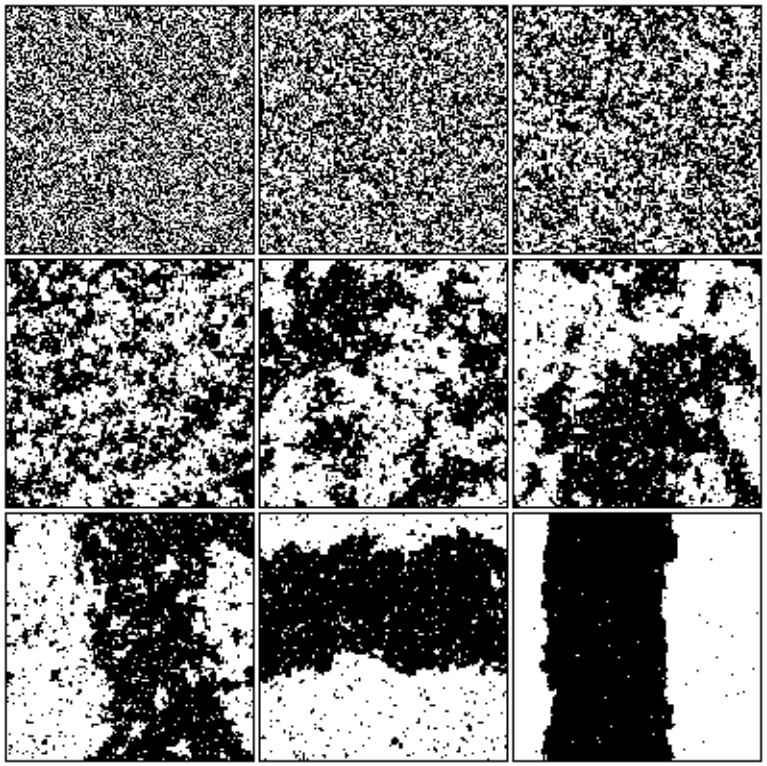

Figure 3.12. Simulations of isotropic first-order Ising models for β-values 0.00, 0.50, 1.00, 1.50, 1.70, 1.76, 1.80, 2.00 and 3.00.

shown simulations of isotropic first-order Ising models for increasing β. The simulations are conditional on 50% of each phase. They are performed using 10000 iterations of the Metropolis spin-flip algorithm described in section 5.3. We see that long-range correlations occur around the critical β.

3.4.2 Morphological extension

In section 3.4.1 only cliques with one or two pixels were concerned. Markov random fields with this restriction are called *pairwise interaction models* or *auto-models* (Besag, 1974). The parameters of a pairwise interaction model will be able to control two very important sets of descriptive features: first-order statistics and second-order statistics. However, these features do not describe all the relevant aspects of a texture. For binary pairwise interaction models we know that we will always have the same structure for the black phase and the white phase, and this does not seem like a natural assumption for many practical purposes. Differences between the two phases can only be controlled using cliques with an odd number of pixels. Ripley (1988) summarized binary images through morphological operations. The study showed that a series of openings and closings made it possible to discriminate between images, where the autocorrelation had little discriminatory power. General surveys on morphological operations can be found in Serra (1982, 1988), Haralick, Sternberg, & Zhuang (1987). We adopt the notation of Haralick et al. (1987) in subsequent morphological expressions.

If we consider the anisotropic first-order model with energy function (3.2), and if we let $C_1(\mathbf{x})$ be the circumference of one of the phases measured by the total number of 0-1-transitions in the image, then we can express the

energy function as

$$U(\mathbf{x}) = -(\alpha + 2\beta)A(\mathbf{x}) + \frac{1}{2}\beta C_1(\mathbf{x}),$$

where $A(\mathbf{x}) = \sum_{i=0}^{n-1} x_i$. Thus the energy is proportional to the circumference. This shows that a reformulation of a model can provide new insight. The energy function (3.2) can also be expressed using the morphological operator *erosion* ($\ominus$), as

$$U(\mathbf{x}) = -\alpha A(\mathbf{x}) - \beta_1 A(\mathbf{x} \ominus B_1) - \beta_2 A(\mathbf{x} \ominus B_2)$$

where

$B_1 =$  $, B_2 =$ 

and $\mathbf{x} \ominus B$ means erosion of the 1-phase of $\mathbf{x}$ with structuring element B.

We will now reformulate the binary Markov random fields on the basis of mathematical morphology. In general the energy function will have the form:

$$U(\mathbf{x}) = -\alpha A(\mathbf{x}) - \sum_{i=1}^{f} \beta_i A(\mathbf{x} \ominus B_i) \tag{3.6}$$

where the structuring elements $\{B_i, i = 1, .., f\}$ can be chosen arbitrarily.

We shall then turn to the formulation of the conditional probabilities. Let $\mathbf{x}_{i,k} = (x_0, x_1, .., x_{i-1}, k, x_{i+1}, .., x_{n-1})$ then

$$\frac{P(X_i = 1 \mid rest)}{P(X_i = 0 \mid rest)} = \frac{P(\mathbf{x}_{i,1})}{P(\mathbf{x}_{i,0})} = \exp(-U(\mathbf{x}_{i,1}) + U(\mathbf{x}_{i,0}))$$

$$= \exp(\alpha + \sum_{j=1}^{k} \beta_j [A(\mathbf{x}_{i,1} \ominus B_j) - A(\mathbf{x}_{i,0} \ominus B_j)])$$

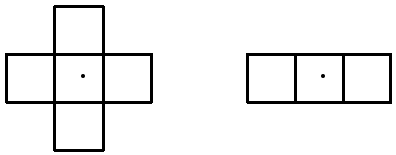

Figure 3.13. Isotropic and anisotropic structuring elements.

$$= \exp(\alpha + \sum_{j=1}^{k} \beta_j n_{i,1}(B_j)).$$

Thus when computing the conditional probabilities we consider the pixels overlapped by B_j placed at pixel i. $n_{i,1}(B_j)$ is defined as the number of these pixels that are members of $\mathbf{x}_{i,1} \ominus B_j$. The computation of the conditional probabilities is local, and this is a very important property for the model to be computationally feasible.

Two interesting structuring elements are shown in figure 3.13. The isotropic element can be used to model isotropic differences between the two phases, and the anisotropic can be used to model anisotropic differences between the two phases. If we let $C_2(\mathbf{x})$ be the circumference of the 1-phase measured by the total number of 1-pixels with a neighboring 0-pixel, then the MRF defined through this measure is equivalent to a model with the isotropic structuring element.

The figures 3.14 to 3.27 show some examples of simulated samples from MRFs with the structuring elements of figure 3.13. The simulations were run on a 128×128 toroidal grid. All samples have approximately 50% black and 50% white pixels. The parameters β_1 and β_2 correspond to the structuring elements of figure 3.13. All the parameter sets, except the one

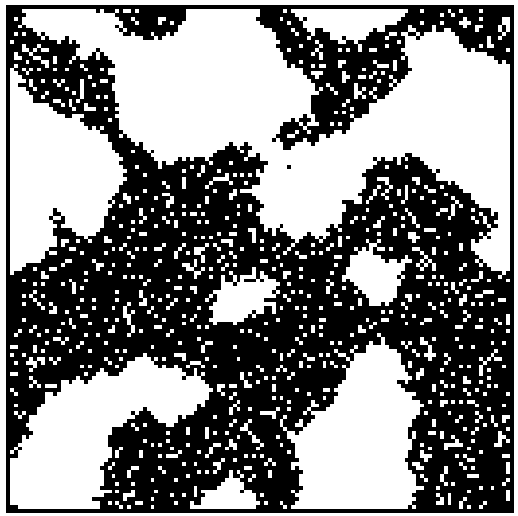

Figure 3.14. Morphological MRF. $\beta_1 = 2.0$, $\beta_2 = 0.0$.

used in figure 3.27, are supercritical. The supercritical samples shown are thus intermediate steps towards some relatively uninteresting steady-state pattern. 50 iterations (full sweeps) of the algorithm were used to create these figures. In all the examples we see a structural difference between the two phases. We have white dots in the black phase but no black dots in the white phase. The structural difference is also reflected in the larger structures. In some of the images there is visually no doubt that it is white objects enclosed in a black phase. Such a difference between the two phases is simply not possible with binary pairwise interaction models.

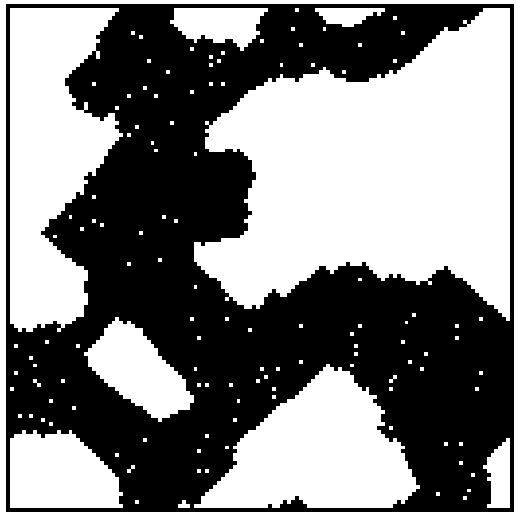

Figure 3.15. Morphological MRF. $\beta_1 = 4.0$, $\beta_2 = 0.0$.

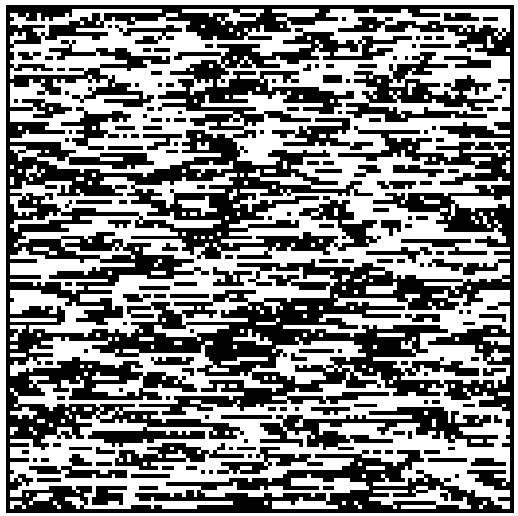

Figure 3.16. Morphological MRF. $\beta_1 = 0.0$, $\beta_2 = 2.0$.

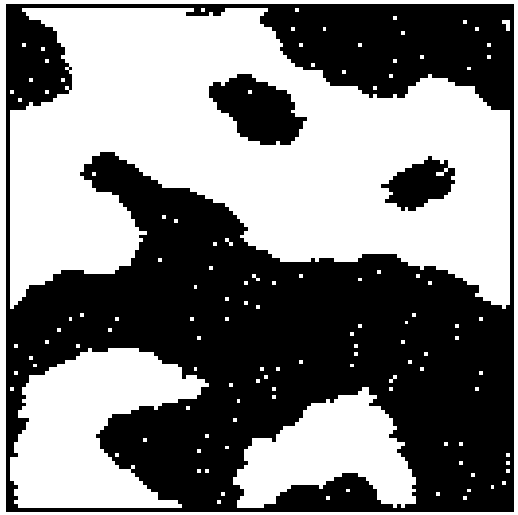

Figure 3.17. Morphological MRF. $\beta_1 = 2.0$, $\beta_2 = 2.0$.

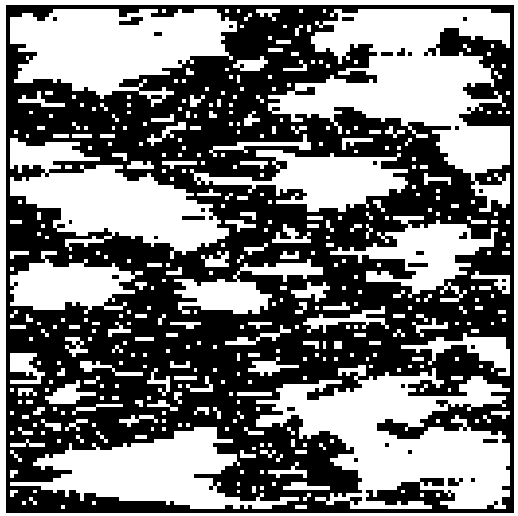

Figure 3.18. Morphological MRF. $\beta_1 = 0.5$, $\beta_2 = 2.0$.

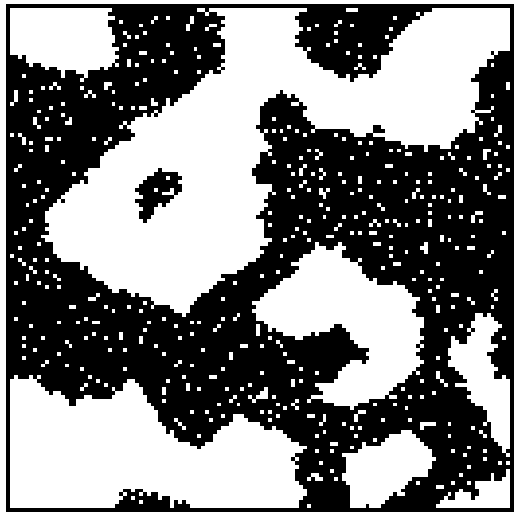

Figure 3.19. Morphological MRF. $\beta_1 = 2.0$, $\beta_2 = 0.5$.

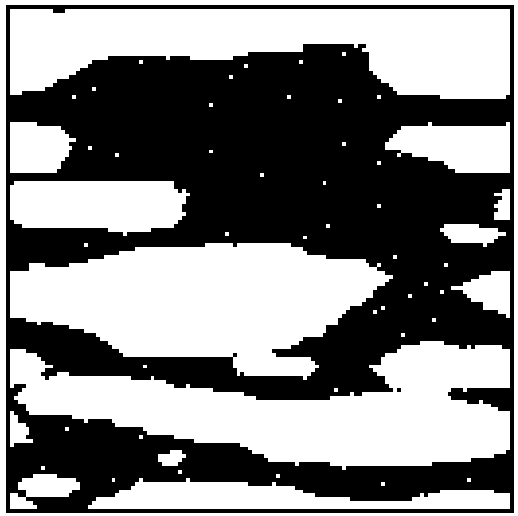

Figure 3.20. Morphological MRF. $\beta_1 = 2.0$, $\beta_2 = 4.0$.

Figure 3.21. Morphological MRF. $\beta_1 = 4.0$, $\beta_2 = 2.0$.

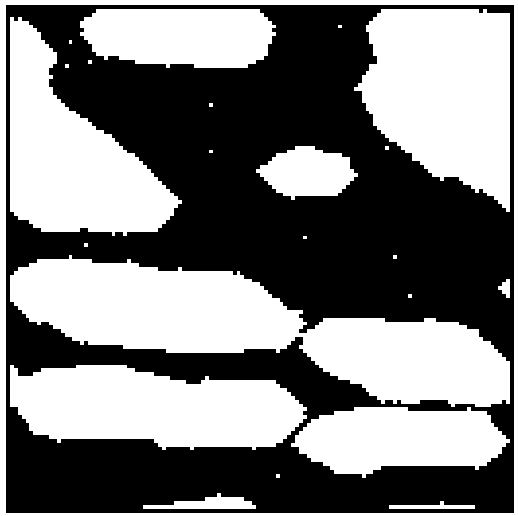

Figure 3.22. Morphological MRF. $\beta_1 = 4.0$, $\beta_2 = 4.0$.

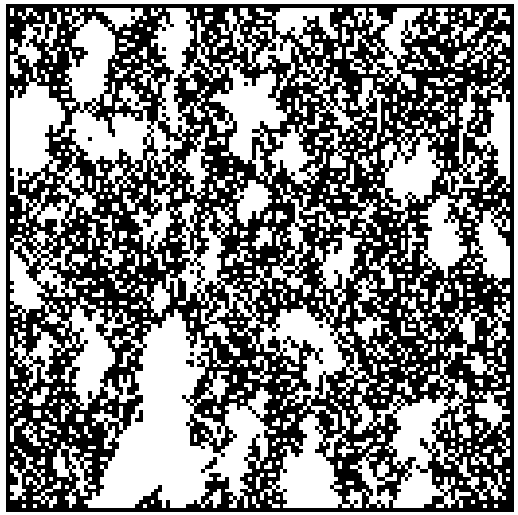

Figure 3.23. Morphological MRF. $\beta_1 = 2.0$, $\beta_2 = -1.0$.

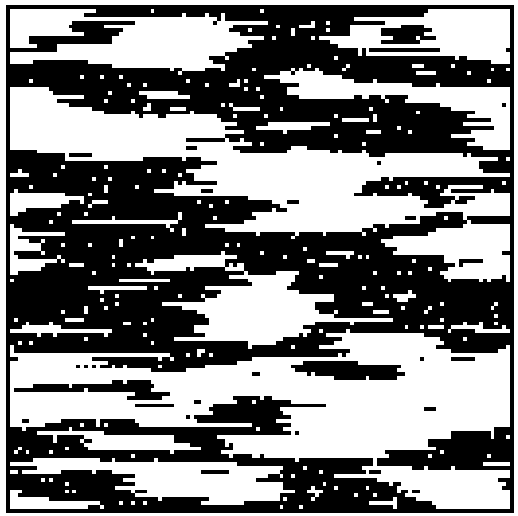

Figure 3.24. Morphological MRF. $\beta_1 = 0.3$, $\beta_2 = 3.0$.

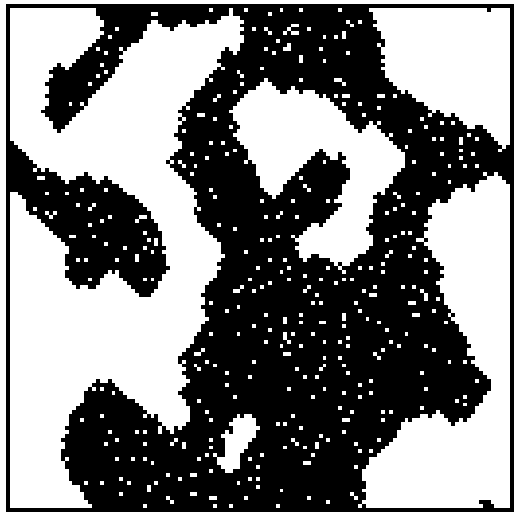

Figure 3.25. Morphological MRF. $\beta_1 = 4.0$, $\beta_2 = -1.0$.

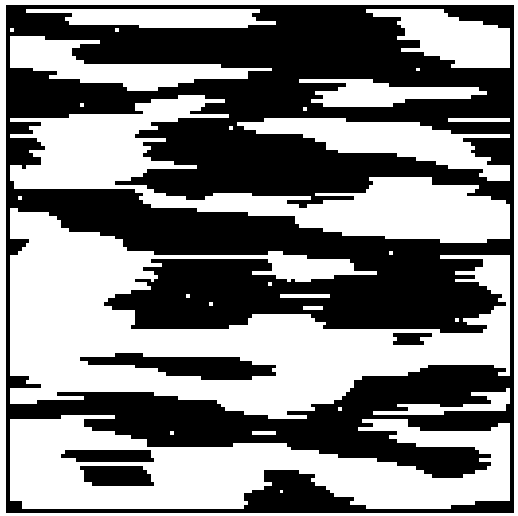

Figure 3.26. Morphological MRF. $\beta_1 = 0.5$, $\beta_2 = 6.0$.

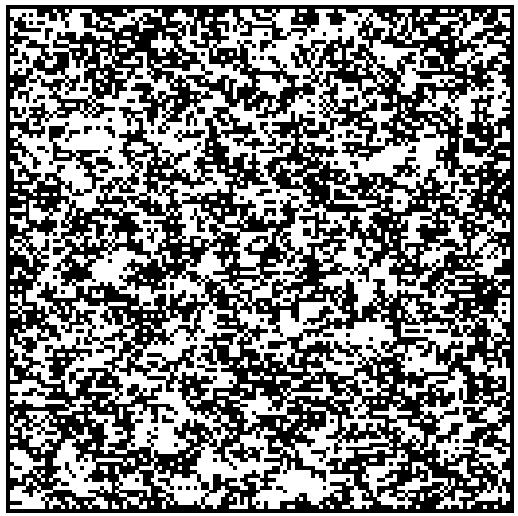

Figure 3.27. Morphological MRF. $\beta_1 = 0.5$, $\beta_2 = 0.5$.

Why formulate morphological MRFs ?

The energy function (3.6) is only a reformulation of the energy function defined through cliques. To every structuring element there is a corresponding clique with the same shape. We propose this reformulation because it brings coherence between the statistical models and descriptive image analysis. It makes it more obvious when to use multi-spin cliques and which it should be. It provides us with the ever increasing toolbox of morphological image analysis as modelling tools. Other energy functions than (3.6) with a more intricate relation between structuring elements and and cliques may be formulated in simple morphological terms.

3.5 Potts models

The Potts model is a generalization of the Ising model to more than two unordered states (phases). It has been studied in statistical physics since it was introduced in Potts (1952). A tutorial review of the results of this research can be found in Wu (1982). A review of the Potts models in a statistical setting can be found in Besag (1986). We shall now review three examples of Potts models. They are of increasing complexity.

Let q be the number of states and $\{1, 2, .., q\}$ the corresponding pixel values. Further let

$$\delta(x_1, x_2, .., x_k) = \begin{cases} 1 & \text{if } x_1 = x_2 = ... = x_k \\ 0 & \text{otherwise} \end{cases}$$

then the standard nearest-neighbor Potts model is characterized by the joint distribution

$$P(\mathbf{x}) = \frac{1}{Z} \exp(\beta \sum_{i \sim j} \delta(x_i, x_j)) \tag{3.7}$$

where

$$Z = \sum_{\Omega} \exp(\beta \sum_{i \sim j} \delta(x_i x_j)).$$

This corresponds to nearest-neighbor cliques having the potential $-\beta$ if the two pixels belong to the same state and zero otherwise. For the conditional probabilities we get

$$P(X_i = k \mid x_j, j \in \mathcal{N}_i) = \frac{\exp(\beta u_i(k))}{\sum_l \exp(\beta u_i(l))}$$

where $u_i(k)$ is the number of neighbors of pixel i with value k. If this Potts model has two states it is equivalent to an isotropic zero-field Ising model, when β from the Potts model is multiplied by two.

The Potts model above can be extended by allowing each state to have a specific structure and frequency of occurrence. This is easily done by introducing state-dependent parameters for neighbor-pairs , $\{\beta_k, k = 1, .., q\}$, and for single pixels $\{\alpha_k, k = 1, .., q\}$, thus obtaining the conditional probabilities

$$P(X_i = k \mid x_j, j \in \mathcal{N}_i) = \frac{\exp(\alpha_k + \beta_k u_i(k))}{\sum_m \exp(\alpha_m + \beta_m u_i(m))}. \tag{3.8}$$

Some ordering between the states can be obtained by letting the parameters be specific for the colors of both neighbors, giving

$$P(X_i = k \mid x_j, j \in \mathcal{N}_i) = \frac{\exp(\alpha_k - \sum_{l \neq k} \beta_{kl} u_i(l))}{\sum_m \exp(\alpha_m - \sum_{l \neq m} \beta_{ml} u_i(l))}. \tag{3.9}$$

3.5.1 Phase transitions

For the q-state Potts model we have phase transitions as those described for the Ising model (Potts, 1952). The critical value of β, β_c, for the model (3.7) is

$$\beta_c = \log(1 + \sqrt{q})$$

and for the 2-state Potts model this gives

$$\beta_c = \log(1 + \sqrt{2}) \Leftrightarrow$$

$$\sinh \beta_c = 1.$$

Thus $\beta_c = 0.8814$.

3.5.2 Morphological extension

It is possible to include multi-spin cliques to incorporate morphological properties in the models. We generalize the notation from the last section by first defining a series of binary images, $\{\mathbf{x}(k), k = 1, .., q\}$, from the q-state image, $\mathbf{x}$, i.e.

$$\mathbf{x}(k) = \begin{cases} 1 & \text{if } x_i = k \\ 0 & \text{otherwise} \end{cases}$$

We can now introduce a morphological Potts model as

$$U(\mathbf{x}) = \sum_{k=1}^{q} [-\alpha_k A(\mathbf{x}(k)) - \sum_{i=1}^{f} \beta_{ik} A(\mathbf{x}(k) \ominus B_{ik})]. \tag{3.10}$$

Examples of this model and its application will be shown in section 5.4.

3.5.3 Other extensions

The literature of statistical physics (Wu, 1982) provides us with some other extensions of the Potts model.

- Site-diluted Potts model
  This model includes vacancies on the grid. These vacancies can be chosen at random or in a deterministic way. Examples of a site-diluted Potts model model and its application will be shown in section 5.4.
- Bond-diluted Potts model
  In this model we allow neighbors with no interaction (or bond). The missing bonds can be chosen at random or in a deterministic way.

- Random-bond Potts model
  In this model the potential of each bond is chosen independently from some probability distribution.
- "Spin-glass" Potts model
  An extension of the binary spin-glass model. The potentials of the bonds is another random field (usually Gaussian).

3.6 Gaussian Markov random fields

The Gaussian Markov random field model is frequently used to describe continuous phenomena. The conditional density is given by the expression

$$P(x_i \mid x_j, j \in \mathcal{N}_i) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\{-\frac{1}{2\sigma^2}[x_i - \mu - \sum_{j \in \mathcal{N}_i} \theta_j (x_j - \mu)]^2\}. \tag{3.11}$$

This model is also called a conditional autoregressive (CAR) model. More detailed descriptions of this model can be found in Besag (1974), Ripley (1981) and Chellappa (1985).

To specify the joint distribution of the CAR model let $\mathbf{B}$ be a $n \times n$ matrix with unit diagonal entries and off-diagonal elements $\{-\theta_{ij}, i \neq j\}$, where $\theta_{ij} = 0$ unless i and j are neighbors. When i and j are neighbors, θ_{ij} equals the θ that corresponds to the relative positions of these two pixels. Thus if the model is defined on a toroidal grid, then $\mathbf{B}$ will be block circulant with circulant blocks; see e.g. Chellappa (1985) or Dubes & Jain (1989). Obviously $\mathbf{B}$ is symmetric. The joint distribution is then multivariate normal with mean vector μ, dispersion matrix $\sigma^2\mathbf{B}^{-1}$ and density

$$f(\mathbf{x}) = \frac{1}{\sqrt{2\pi\sigma^2}^n} \sqrt{\mid \mathbf{B} \mid} \exp\{-\frac{1}{2\sigma^2}(\mathbf{x} - \mu)^T \mathbf{B} (\mathbf{x} - \mu)\}. \tag{3.12}$$

For this model to be valid we have to require that $\mathbf{B}$ is positive definite.

The CAR models are related to the simultaneous autoregressive (SAR) models (Besag, 1974; Ripley, 1981; Kashyap & Chellappa, 1983). SAR models are extensions of the autoregressive models of time series analysis to two dimensions.

3.6.1 Alternative gray level distributions

The joint density in equation 3.12 corresponds to the energy function

$$U(\mathbf{x}) = -\sum_{i\sim j} \beta_{ij} \frac{(x_i - \mu)(x_j - \mu)}{\sigma^2}. \tag{3.13}$$

Besag (1989) presents an alternative class of joint distributions, where the energy function involves pairwise differences only. They are defined by

$$U(\mathbf{x}) = \sum_{i\sim j} \phi(x_i - x_j) \tag{3.14}$$

where ϕ is a function that satisfies

$$\phi(z) = \phi(-z), \phi(z) \text{ increasing with } |z|.$$

Joint distributions defined by equation 3.14 are improper in that they can not be normalized (Besag, 1989). They do however have a perfectly proper conditional density

$$p(x_i|x_j, j \neq i) \propto \exp\{-\sum_{j\in\mathcal{N}_i} \phi(x_i - x_j)\}.$$

It is possible to formulate morphological alternatives to the energy function (3.13) using the operators of gray level morphology (Sternberg, 1986; Haralick et al., 1987). Such models may turn out to be feasible and useful.

Chapter 4

Markov random field parameter estimation

For most practical applications of Markov random fields it is essential that we have accurate and feasible algorithms for parameter estimation. This chapter reviews a selection of estimation methods. Some of these methods are applied in chapter 5. An extension of the asymptotic maximum likelihood estimator (Pickard, 1987) to the anisotropic case is proposed in section 4.4.2.

4.1 Introduction

Maximum likelihood estimation of the MRF parameters is in general computationally intractable due to the likewise intractable partition function in the joint probability density. There are however, as we shall see, exceptions to this rule. But first we will describe some alternatives to ML-estimation.

4.2 Coding estimation

Besag (1974) introduced *coding estimation* as an alternative to ML-estimation. The grid is partitioned into a number of disjoint set of pixels, called *coding patterns*. The codings are chosen such that the distribution of the pixel values within one coding pattern, conditional on the pixel values of the other coding patterns, are independent. This simply means that a pixel and its neighbor cannot be members of the same coding pattern. The number of coding patterns is kept as low as possible to obtain the most efficient estimator. Thus we get two coding patterns for a first-order MRF and four coding patterns for a second-order MRF. These coding patterns are shown in figure 4.1 and figure 4.2 respectively. Since the variables associated with pixels from one coding pattern are conditionally independent, given the observed values of all other pixels, we can express the conditional likelihood as

$$L_k = \prod_{i \in C_k} P(x_i \mid x_j, j \in \mathcal{N}_i)$$

where C_k is the set of pixels belonging to coding pattern k. We get one set of estimates for each coding pattern by maximizing the corresponding

1	2	1	2	1	2
2	1	2	1	2	1
1	2	1	2	1	2
2	1	2	1	2	1
1	2	1	2	1	2
2	1	2	1	2	1

Figure 4.1. *Coding patterns for a first-order MRF. Pixels with the same number belong to the same coding pattern.*

1	2	1	2	1	2
3	4	3	4	3	4
1	2	1	2	1	2
3	4	3	4	3	4
1	2	1	2	1	2
3	4	3	4	3	4

Figure 4.2. *Coding patterns for a second-order MRF. Pixels with the same number belong to the same coding pattern.*

likelihood function. These sets may then be combined appropriately, e.g. by computing the arithmetic or harmonic mean.

4.3 Pseudolikelihood estimation

Besag (1975) suggested using the product of conditional probabilities for all pixels as a *pseudolikelihood* function, i.e. parameter estimates were found by maximizing

$$PL = \prod_i P(x_i \mid x_j, j \in \mathcal{N}_i).$$

This is obviously not a real likelihood function because the conditional probabilities are not independent. Geman & Graffigne (1987) showed however that this method produced consistent estimates in the large graph limit under mild conditions. The reason for using maximum pseudolikelihood estimation instead of coding estimation is to increase the efficiency. Maximum pseudolikelihood estimates compare favorably to coding estimates in Besag (1977), where Gaussian MRFs are considered. Besag (1977) also noted that for the first-order Gaussian Markov random field on a square grid the maximum pseudolikelihood estimator is equivalent to the harmonic mean of the two alternative coding estimator. In the subsequent technical description of estimators for specific models the coding method will give results similar to the pseudolikelihood method.

4.4 Binary MRF

4.4.1 Maximum pseudolikelihood

In this section we present the results for the binomial MRF because these are immediate extensions of the results for the binary MRF.

Let β be the vector of MRF parameters and $\mathbf{s}_i$ be the vector of the corresponding neighbor sums for pixel i, i.e. for a anisotropic first-order MRF we have

$$\beta = \begin{pmatrix} \alpha \\ \beta_1 \\ \beta_2 \end{pmatrix} \quad , \quad \mathbf{s}_i = \begin{pmatrix} 1 \\ x_W + x_E \\ x_N + x_S \end{pmatrix} .$$

Further let

$$T_i = \beta^T \mathbf{s}_i$$

and

$$p_i = \frac{\exp(T_i)}{1 + \exp(T_i)}.$$

Then we can express the conditional distributions of a binomial MRF as

$$(x_i | x_j, j \neq i) \in B(n, p_i).$$

For a binary MRF n will be equal to one.

For the binomial MRF the conditional probability of an observed pixel value given the rest of the observed image is

$$P(x_i \mid x_j, j \in \mathcal{N}_i) = \begin{pmatrix} n \\ x_i \end{pmatrix} p_i^{x_i} (1 - p_i)^{n - x_i}.$$

The resulting pseudolikelihood is

$$PL = \prod_i \begin{pmatrix} n \\ x_i \end{pmatrix} p_i^{x_i} (1 - p_i)^{n - x_i}.$$

Thus we maximize

$$\log PL = \sum_i [\log \begin{pmatrix} n \\ x_i \end{pmatrix} + x_i T_i - n \log(1 + \exp(T_i))].$$

The binomial coefficient does not depend on β thus the maximum pseudolikelihood estimate of β is found by maximizing

$$f(\beta) = \sum_i [x_i T_i - n \log(1 + \exp(T_i))] \tag{4.1}$$

with respect to β. For this function we can find the gradient vector $\nabla f(\beta)$ and Hessian matrix as $\nabla^2 f(\beta)$

$$\nabla f(\beta) = \sum_i [x_i - n p_i] \mathbf{s}_i$$

$$\nabla^2 f(\beta) = -n \sum_i \frac{\exp(T_i)}{(1 + \exp(T_i))^2} \mathbf{s_i s_i}^T.$$

The Hessian matrix is negative semi-definite, and the maximization problem is now easily solved by standard optimization procedures.

Dubes & Jain (1989) expresses the concern that when maximizing the function f in (4.1) we may run into a local maximum. This requires that the optimization is repeated for several initial guesses. However, we have experienced that we obtain the same solution from several initial guesses, and that for simulated textures this solution correspond to the parameters used in the simulation. The function f seems to be well-behaved even for real textures. In figure 4.3 we show $f(\alpha, \beta)$ for an isotropic first-order MRF estimated on a binary grass lawn texture (Brodatz texture D9).

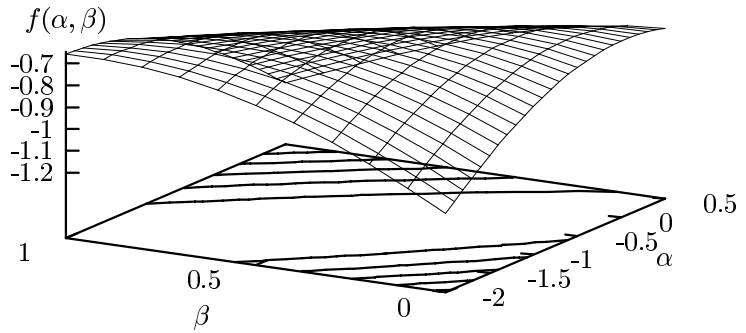

Figure 4.3. Pseudolikelihood surface for binary grass. Maximum is reached for $\alpha = -1.27$ and $\beta = 0.64$. The lines in the (α,β) plane are isolines for f.

4.4.2 Asymptotic maximum likelihood

In the case of a zero-field first-order binary MRF we can use the results of Onsager (1944) for the large-grid limit to estimate β. The method was introduced by Pickard (1987) for the isotropic case. He used equation 3.5 and applied it to a finite grid. Thus in our notation he got the equation

$$Corr(x_i, x_j | i \sim j) = \frac{2}{N} \sum_{i \sim j} x_i x_j - 1 = \Gamma(\beta).$$

The equation can be solved numerically using e.g. *Brent's method* (See e.g. Press et al. (1988)). For grids larger than 100×100 Pickard showed that the finite-grid gamma-functions are nearly identical. The results can be extended to the anisotropic case, using equation 3.4 and the vertical analogue. We get the equations

$$Corr(x_i, x_j | i \leftrightarrow j) = \frac{4}{N} \sum_{i \leftrightarrow j} x_i x_j - 1 = \Gamma_1(\beta_1, \beta_2)$$

$$Corr(x_i, x_j | i \updownarrow j) = \frac{4}{N} \sum_{i \updownarrow j} x_i x_j - 1 = \Gamma_2(\beta_1, \beta_2)$$

Solving these two equations will provide us with estimates of β_1 and β_2. The solution can be found using a Newton-Raphson method (See e.g. Press et al. (1988)).

4.4.3 Other estimation methods

Derin & Elliot (1987) introduced an alternative estimation method that involves the solution of an overdetermined system of linear equations. This and other ad hoc methods are reviewed in Dubes & Jain (1989).

4.5 Potts model

In this section we consider the q-state Potts model with conditional probability defined in (3.8).

4.5.1 Maximum pseudolikelihood

The conditional probabilities are given by

$$p_i(k) = \frac{\exp(\alpha_k + \beta_k u_i(k))}{\sum_m \exp(\alpha_m + \beta_m u_i(m))}$$

where, as before, $u_i(k)$ is the number of neighbors of pixel i with value k.

Let β be the vector of Markov parameters and $\mathbf{s}_i(k)$ be the vector of the corresponding neighbor functions for pixel i and color k, i.e.

$$\beta = \begin{pmatrix} \alpha_1 \\ \beta_1 \\ \vdots \\ \alpha_q \\ \beta_q \end{pmatrix} \quad , \quad \mathbf{s}_i(k) = \begin{pmatrix} 1 \\ u_i(k) \end{pmatrix} .$$

The pseudolikelihood function

$$PL = \prod_i p_i(x_i)$$

is then maximized by maximizing

$$f(\beta) = \sum_i [\alpha_{x_i} + \beta_k u_i(x_i) - \log \sum_m \exp(\alpha_m + \beta_m u_i(m))].$$

The gradient vector is easily obtained as

$$\nabla f(\beta) = \begin{pmatrix} \sum_i [1_{x_i=1} - p_i(1)]\mathbf{s}_i(1) \\ \sum_i [1_{x_i=2} - p_i(2)]\mathbf{s}_i(2) \\ \vdots \\ \sum_i [1_{x_i=q} - p_i(q)]\mathbf{s}_i(q) \end{pmatrix}$$

However, if a constant is added to every α_k we get exactly the same model. Thus one α_k can be chosen arbitrarily and we then remove the corresponding equation above.

4.6 Gaussian MRF

This section describes to ways of estimating parameters of the Gaussian Markov random field model defined in section 3.6.

4.6.1 Maximum pseudolikelihood

From the conditional distribution given by equation (3.11) we find that the the pseudolikelihood function is given by

$$PL = \prod_i \frac{1}{\sqrt{2\pi\sigma^2}} \exp\{-\frac{1}{\sigma^2}(x_i - \mu - \sum_{j \in \mathcal{N}_i} \theta_j x_j)^2\}.$$

Let θ be the vector of parameters and $\mathbf{s}_i$ be the vector of the corresponding neighbor sums for pixel i, i.e.

$$\theta = \begin{pmatrix} \mu \\ \theta_1 \\ \theta_2 \\ \vdots \\ \theta_r \end{pmatrix} \quad , \quad \mathbf{s}_i = \begin{pmatrix} 1 \\ x_W + x_E \\ x_N + x_S \\ \vdots \\ x_U + x_V \end{pmatrix} .$$

By setting the partial derivatives of the log-likelihood equal to zero we obtain

$$\hat{\theta} = [\sum_i \mathbf{s_i s_i}^T]^{-1} \sum_i \mathbf{s}_i x_i$$

and

$$\hat{\sigma}^2 = \frac{1}{N} \sum_i (x_i - \hat{\theta}^T \mathbf{s}_i)^2$$

$$= \frac{1}{N} (\sum_{(i,j)} x_{i,j}^2 - \hat{\theta}^T \sum_{(i,j)} \mathbf{s} x_i).$$

Thus the solution of the estimation problem is given in closed form.

4.6.2 Maximum likelihood

The joint distribution given by equation (3.12) provides us with the likelihood function

$$L = \frac{1}{\sqrt{2\pi\sigma^2}^n} \sqrt{|\,\mathbf{B}\,|} \exp\{-\frac{1}{2\sigma^2} (\mathbf{x} - \mu)^T \mathbf{B} (\mathbf{x} - \mu)\}. \qquad (4.2)$$

Let us assume (Besag, 1974) that $\mu = \mathbf{0}$ and that we have an estimate of $\mathbf{B}$, $\hat{\mathbf{B}}$. Then the ML-estimate of σ^2 will be

$$\hat{\sigma}^2 = \frac{1}{n} \mathbf{x}^T \hat{\mathbf{B}} \mathbf{x}$$

and by substituting this into equation 4.2 and taking the logarithm leads us to finding the ML-estimate by maximizing

$$\log |\mathbf{B}| - n \log \mathbf{x}^T \mathbf{B} \mathbf{x}.$$

We are now left with the numerical problem of evaluating this function and especially the determinant $|\mathbf{B}|$. This has been tried in e.g. Besag & Moran (1975) and Kashyap & Chellappa (1983).

Chapter 5

Markov random field simulation

In this chapter we review a set of iterative simulation schemes for Markov random field simulation. We then present a fast new parallel algorithm for simulating Markov random fields conditional on given first-order statistics. Finally we investigate the use of this algorithm and a morphological Potts model in the simulation of geological structures.

5.1 Introduction

The problem of generating samples from a MRF distribution is important for a number of reasons. Obviously in image analysis we are concerned with the visual properties of the samples. In statistical physics it is more interesting to use the samples for computing expected values of thermodynamic quantities.

If we disregard the spatial nature of image data and consider the pixel values as identically and independently distributed then the pixel value histogram will be a sufficient statistic for our random field. Simulating an image from the first-order statistics would only require sampling from a univariate distribution which is relatively easy but rather uninteresting. Simulating a more general random field corresponds to sampling a multivariate distribution of very high dimension, and a selection of iterative simulation schemes has been developed (See e.g. Dubes & Jain (1989)).

5.2 Iterative simulation

The iterative process of MRF simulation has fruitfully been thought of as a discrete, finite-state Markov chain. The state-space of this Markov chain is the set of all possible configurations Ω and the limiting distribution we want is the MRF distribution.

From the theory of discrete, finite-state Markov chains we get the following definitions and results. Let $\mathbf{P} = \{p_{ij}, i, j \in \Omega\}$ be the matrix of transition

probabilities, where $p_{ij}(t)$ denotes the probability of a transition from state i to state j in t steps.

Definition 5 . *A Markov chain is* **irreducible** *or* **non-decomposable** *iff*

$$\forall i, j \in \Omega \; \exists t : p_{ij}(t) > 0$$

Definition 6 *A Markov chain is* **aperiodic** *iff*

$$\exists t_0 \forall t > t_0 \forall i, j \in \Omega : p_{ij}(t) > 0$$

Lemma 1 . *An irreducible Markov chain is aperiodic if*

$$\exists i \in \Omega : p_{ii} > 0$$

Proof. See Aarts & Korst (1989).

Definition 7 . *A probability distribution π is* **invariant** *or* **stationary** *for a Markov chain with transition probabilities $\{p_{ij}\}$ iff the* **global balance equations** *are satisfied, that is*

$$\forall j \in \Omega : \pi_j = \sum_i \pi_i p_{ij}$$

Theorem 2 . *For an irreducible and aperiodic Markov chain there exists a unique invariant distribution.*

Proof. See e.g. Feller (1968).

Definition 8 . *A Markov chain is* **reversible** *or* **self-adjoint** *iff the* **detailed balance equations** *are satisfied, that is*

$$\forall i, j \in \Omega : \pi_i p_{ij} = \pi_j p_{ji}$$

Lemma 2 . *For an irreducible and aperiodic Markov chain π is the unique invariant distribution if it satisfies the detailed balance equations.*

Proof. See e.g. Aarts & Korst (1989).

5.2.1 The Metropolis algorithm

Metropolis, Rosenbluth, Rosenbluth, Teller, & Teller (1953) described an algorithm for computer simulation of Gibbs distributed systems. This algorithm is now known as the *Metropolis algorithm.*

Algorithm 1 . **Metropolis algorithm**. *Let* $\mathbf{Q}$ *be a symmetric irreducible transition matrix with state space* Ω.

1. *Start with configuration* $\mathbf{x} \in \Omega$
2. *Choose a new configuration* $\mathbf{y}$ *from the distribution in the row corresponding to* $\mathbf{x}$ *in* $\mathbf{Q}$
3. *Replace* $\mathbf{x}$ *by* $\mathbf{y}$ *with probability*

$$p = min(1, P(\mathbf{X} = \mathbf{y})/P(\mathbf{X} = \mathbf{x}))$$

4. *if not stop then goto 2*

Notice that while the Metropolis algorithm will always make a change to a new configuration with higher probability it will also with some probability make a change to a new configuration with lower probability.

It is trivial to show that the Metropolis algorithm defines an irreducible and aperiodic Markov chain. The detailed balance equations give for $i \neq j$

$$q_{ij}p_i \min(1, \frac{p_j}{p_i}) = q_{ji}p_j \min(1, \frac{p_i}{p_j})$$

which for both $p_i \geq p_j$ and $p_i < p_j$ leads to

$$q_{ij} = q_{ji}$$

This explains the symmetry condition on $\mathbf{Q}$.

5.2.2 Spin-flip algorithms

In spin-flip algorithms single pixels are visited successively and their values are changed according to some criteria. Kirkland (1989) considered flipping 2x2 and 3x3 blocks of pixels but the results were not encouraging. The two most popular flipping criteria provides the following algorithms.

Algorithm 2 . **Metropolis spin-flip algorithm**. *Let* $\mathbf{Q}$ *be a symmetric irreducible transition matrix with state space* $\{0, .., G-1\}$*, where* G *is the number of gray levels.*

1. *Start with configuration* $\mathbf{x}$
2. *Choose a pixel* s *and a pixel value* g *from the distribution in the row corresponding to* x_s *in* $\mathbf{Q}$.

3. *Set configuration* $\mathbf{y}$ *equal to* $\mathbf{x}$ *with pixel* s *set to* g
4. *Replace* $\mathbf{x}$ *by* $\mathbf{y}$ *with probability*

$$p = min(1, P(\mathbf{X} = \mathbf{y})/P(\mathbf{X} = \mathbf{x}))$$

5. *if not stop then goto 2*

Algorithm 3 . **Gibbs sampler** *or* **heat bath algorithm**.

1. *Start with configuration* $\mathbf{x}$
2. *Choose a pixel* s
3. *Replace* x_s *by a value sampled from the conditional distribution of* X_s *given the values of the neighbors of* s.
4. *if not stop then goto 2*

In both algorithms we have to choose (visit) a pixel for each iteration. One way could be to choose a random pixel every time, but making a systematic sweep over the image is more efficient both in terms of rate of convergence and in terms of time per sweep. If we make sure that we continue to visit every pixel then the order in which we sweep through the image does not matter. Using a simple raster sweep does ensure convergence but imposes an artificial anisotropy on the intermediate results as seen (unintentionally ?) in the isotropic simulations of figure 4 in Derin & Elliot (1987). To avoid the artificial anisotropy and to enable simultaneous updating of many pixels we divide the image in coding patterns as described in section 4.2. Pixels from each coding pattern do not interact with other pixels from the same coding pattern. We then sweep through the coding patterns, one at a time.

5.2.3 The Metropolis spin-exchange algorithm

The spin-exchange algorithm was introduced in image analysis by Cross & Jain (1983).

Algorithm 4 . Metropolis spin-exchange algorithm.

1. *Start with configuration* $\mathbf{x}$
2. *Choose two pixels* r *and* s *at random*
3. *If* $x_r = x_s$ *then goto 2*
4. *Set* $\mathbf{y}$ *equal to* $\mathbf{x}$ *with pixels* r *and* s *switched*
5. *Replace* $\mathbf{x}$ *by* $\mathbf{y}$ *with probability*

$$p = min(1, P(\mathbf{X} = \mathbf{y})/P(\mathbf{X} = \mathbf{x}))$$

6. *if not stop then goto 2*

In stead of flipping single pixels this algorithm exchanges the values of two randomly chosen pixels. The step will maintain the pixel value histogram and thus the first-order statistics. As a stop-criteria Cross and Jain checked if the number of successful switchings dropped below 1% or the estimated parameters matched the input parameters within 5%. This resulted in a variety of interesting textures.

To elaborate on this algorithm for the isotropic Ising model we consider the case $x_r = 1$ and $x_s = 0$, and thus $y_r = 0$ and $y_s = 1$. The configurations $\mathbf{x}$

and $\mathbf{y}$ are identical except at pixels r and s. The ratio R is then computed as

$$R = \frac{P(\mathbf{Y} = \mathbf{y})}{P(\mathbf{X} = \mathbf{x})} = \frac{\frac{1}{Z(\alpha,\beta)} \exp(\alpha \sum_i y_i + \beta \sum_{i\sim j} y_i y_j)}{\frac{1}{Z(\alpha,\beta)} \exp(\alpha \sum_i x_i + \beta \sum_{i\sim j} x_i x_j)}$$

$$= \exp(\beta \sum_{i\sim j} [y_i y_j - x_i x_j]) = \exp(\beta[W_s(\mathbf{y}) - W_r(\mathbf{x})])$$

where $W_k(\mathbf{z})$ is the number of 1-neighbors of k in configuration $\mathbf{z}$. Ripley (1987) discusses a problem in the exposition of Cross and Jain for the case where r and s are neighbors. This problem does not occur using the present exposition.

The fact that $\sum_i x_i = \sum_i y_i$ means that α is a redundant parameter. This seems quite natural since α is the parameter that controls the relative number of 1-pixels, and this number is kept constant by the spin-exchange algorithm.

5.2.4 Swendsen-Wang algorithm

A relatively new type of simulation algorithm involves flipping clusters. A cluster is a connected set of pixels with identical values. Swendsen & Wang (1987) described a cluster algorithm for the basic Potts model 3.7.

Algorithm 5 . Swendsen-Wang algorithm.

1. *Start with pixel configuration* $\mathbf{x}$

2. *Create a bond configuration by introducing a bond between neighboring pixels with the same color with probability*

$$p = 1 - \exp(-\beta)$$

3. *Find the clusters joined by bonds*

4. *Independently assign a random color to each cluster*

5. *if not stop then goto 2*

Cluster algorithms is an active research area and extended and new versions have occurred (e.g. (Wolff, 1989)).

Figure 5.1 shows 24 iterations of the Swendsen-Wang algorithm on a 5-state Potts model with $\beta = 2.0$. Convergence seems to be very fast. Figure 5.2 is a plot of the maximum pseudolikelihood estimate β as a function of the iteration number.

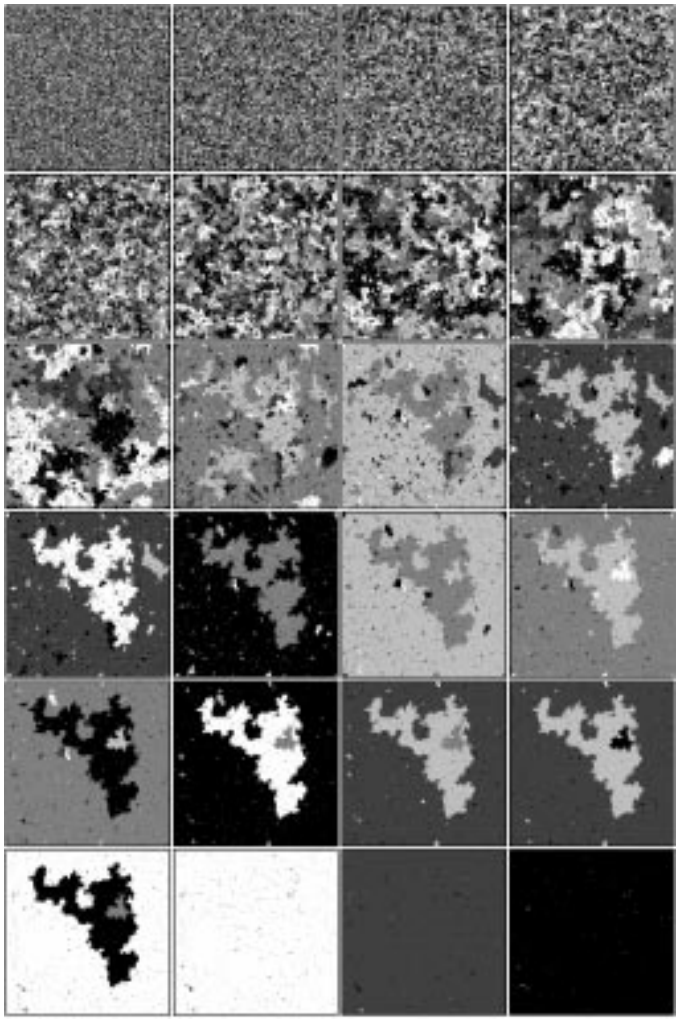

Figure 5.1. 24 iterations of the Swendsen-Wang algorithm on a 5-state Potts model with $\beta = 2.0$.

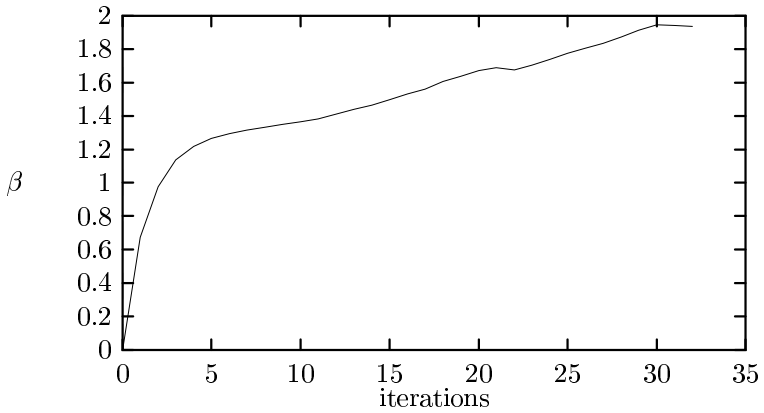

Figure 5.2. Maximum pseudolikelihood estimate β as a function of the iteration number. $\beta = 2.0$ for the simulation.

5.3 The α -controlled spin-flip algorithm

The Metropolis spin-exchange algorithm is the most widely used algorithm for simulating Markov random fields conditional on the first-order statistics. In this section we propose two spin-flip alternatives based on the Gibbs sampler and the Metropolis algorithm and includes as a new feature a feedback loop to achieve the conditioning. The rate of convergence for large attraction parameters β is compared to the rate of convergence of the Metropolis spin-exchange algorithm. The spin-flip algorithms turn out to be faster not only in time per sweep but also in rate of convergence. Further the spin-flip algorithms are easy to 0, and this is done using a SIMD massively parallel computer.

5.3.1 Introduction

Simulating Markov random fields conditional on their first-order statistics has been very popular, since this can provide interesting textures for large β seemingly avoiding the phase transition. The spin-exchange algorithm is however very slow for large β and this is partly due to the fact that the intermediate configurations have fixed first-order statistics and thus the number of possible paths between two configurations are very limited.

In the spin-flip alternatives presented here we do not strictly maintain the first-order statistics, but stabilizes these around a preset value through a feedback loop.

5.3.2 The feedback loop

The idea is to construct a spin-flip algorithm with almost constant first-order statistics. For an Ising model the first-order statistics is fully described by the mean μ. Suppose we want μ to have the value μ_0. This may be accomplished by a feedback loop such that α is adjusted after each iteration to keep $\mu(t)$ near μ_0. In control theory (e.g. Åström & Wittenmark (1984)) the standard textbook PID-controller can be written as

$$\alpha(t) = K_p e(t) + K_i \frac{1}{1-q^{-1}} e(t) + K_d(1-q^{-1})e(t)$$

where $e(t) = \mu_0 - \mu(t)$ is the error function and q^{-1} is the backshift-operator, i.e. $q^{-1}e(t) = e(t-1)$. If we multiply with $(1-q^{-1})$ on both sides we get

$$(1-q^{-1})\alpha(t) = K_p(1-q^{-1})e(t) + K_i e(t) + K_d(1-q^{-1})^2 e(t) \qquad (5.1)$$

and this is the form actually used here. The name, PID-controller, comes from the three controlling actions in expression 5.1.

- P - proportional action. The basic idea is to have a control action proportional to the error.
- I - integral action. This action is used to eliminate a stationary error in the mean value.
- D - derivative action. This action is used to increase the speed of the control system.

The P-, I- and D- actions are adjusted through K_p, K_i and K_d respectively. The joint PID-action and the dynamic of the spin-flip system determines if the control system is stable.

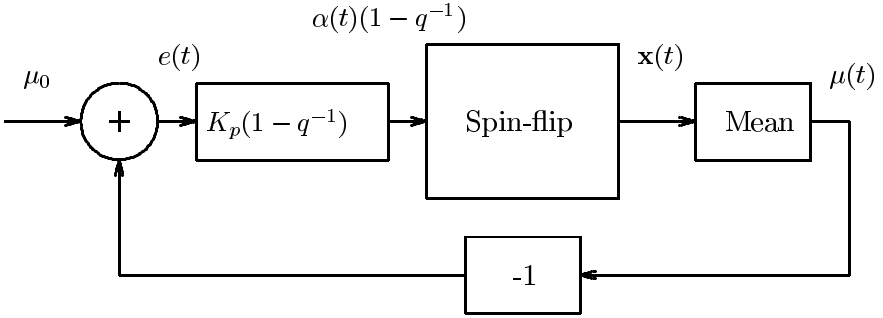

Figure 5.3. *P-controller. Multiplication is performed is the boxes labelled* $K_p(1-q^{-1})$ *and* -1.

A simple P-controller, with $K_i = K_d = 0$, is shown in figure 5.3, and this controller was investigated in the present work. Here α is directly proportional to the error function, $e(t)$. Extension to a PID-controller with integral and derivative actions is possible. This can be done without much computational effort but one has to be more careful in choosing the right constants. To include knowledge about $\mathbf{x}(t-1)$ in computing $\alpha(t)$ would be computationally harder and, as we shall see, completely unnecessary.

The time step used in the control loop is selected as a trade-off between computational cost and performance. The time step used here corresponds to a full sweep of the spin-flip algorithm.

This control loop approach is generally applicable to iterative simulation schemes and is not confined to control of the mean value. Other properties of the configuration $\mathbf{x}$, e.g. second-order statistics, can be measured and used to control the simulation parameters.

5.3.3 Relation to importance sampling

Green (1986) suggested that global properties (e.g. first-order statistics) could be integrated in a Markov random field $P(\mathbf{x})$ by considering the modified field

$$P^*(\mathbf{x}) \propto e^{-\sigma D(\mathbf{x})} P(\mathbf{x})$$

where $D(\mathbf{x})$ is a non-negative random variable measuring the deviation from the ideal property, and σ is a positive parameter. This modified field focuses $P(\mathbf{x})$ on realizations with the desired property. The parameter determines the strength of the focusing. For the Ising model we might choose

$$D(\mathbf{x}) = (n(\mathbf{x}) - n_d)^2$$

where $n(\mathbf{x})$ is the actual number of 1-pixels in $\mathbf{x}$, and n_d is the desired number of 1-pixels. The conditional distribution of pixel i in the modified field is then given by replacing α with $\alpha - 2\sigma(n(\mathbf{x}_{x_i=0}) - n_d + \frac{1}{2})$. This result corresponds to a local P-controller with $K_p = 2\sigma$. We shall adopt the term *importance sampling* from Ripley (1992) for the sampling from $P^*(\mathbf{x})$.

There are three advantages in using the PID-controller in stead of importance sampling. The α-adjustment is done once per sweep in stead of once per pixel. Importance sampling can not be parallelized because the conditional distribution is based on global properties. Finally we have to know the value of α when doing importance sampling, whereas the PID-controller will (hopefully) converge to the correct value from any starting guess.

5.3.4 Parallel implementation

Parallel implementations of MRF simulation schemes has been suggested and implemented several times in the past. Geman & Geman (1984) discussed a parallel implementation of the Gibbs sampler and suggested an asynchronous updating scheme based on an MIMD (multiple instruction multiple data) computer. Murray, Kashko, & Buxton (1986) implemented a parallel version of the Metropolis spin-flip algorithm using synchronous updating for every coding on an SIMD (single instruction multiple data) computer.

An approximation of the Metropolis spin-exchange algorithm was implemented on a SIMD computer by Margalit (1989) through a slave-master handshaking between the two chosen pixels. This procedure was run both with and without using a coding scheme. Besides the handshaking overhead only a 40% degree of parallelism is accomplished.

The α-controlled spin-flip algorithm described here was implemented on a Connection Machine CM-200 from Thinking Machines using the parallel C-compiler, C*. The basic shape of the parallel variables correspond to a grid of *coding elements*. A coding element is a group of neighboring pixels, one from each coding. The coding element for a first-order MRF could be a pixel from coding 1 and it's neighbor to the right. For a second-order MRF the coding element could be a 2x2 square with a pixel from coding 1 in it's upper left corner. We associate a virtual processor with each coding element and this processor performs the spin-flip operation successively on all the pixels in that coding element. If the number of coding elements is equal to a multiple of the number of physical processors this scheme provides 100% use of the parallel computer.

5.3.5 Results

The simulations were performed on a toroidal 128 by 128 grid. One iteration corresponds to a full sweep through the image.

Mean-convergence

In order to make the mean value converge to the desired μ_0 we have to choose an appropriate constant K_p. We started simulations with all black pixels and μ_0 set to 0.5. Different values of K_p were chosen for both Metropolis algorithm and the Gibbs sampler and the results are shown in figure 5.4 and figure 5.5. β was 3.0 for all the curves. We can see that $K_p = 6.0$ seems like an appropriate choice for both algorithms. We also notice that ringing effects are more apparent for the Metropolis algorithm. This is due to the more frequent flipping.

β-convergence

The rate of convergence of the pseudo-likelihood β-estimate is shown in figure 5.6. The value of β is 3.0 and we can see that the spin-exchange algorithm converges slower than it's spin-flip alternatives. To illustrate the visual convergence of the three algorithms we show typical configurations after 50, 100, 200, 400, 800, 2000, 4000 and 8000 iterations for $\beta = 3.0$. In figure 5.7 we see that the Metropolis spin-flip has converged to a stable pattern already after 200 iterations. The convergence of the Gibbs sampler is shown in figure 5.8 and a stable pattern is reached after 2000 iterations.

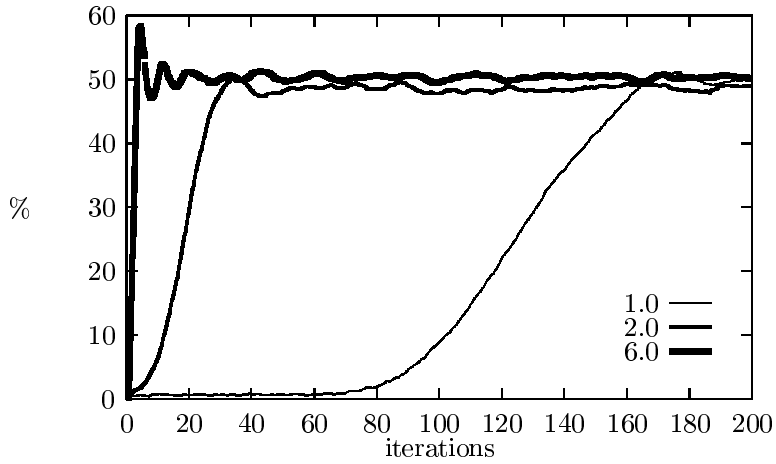

Figure 5.4. *Mean-convergence for Metropolis spin-flip. Percentage of 1-pixels versus the number of iterations for* $\beta = 3.0$.

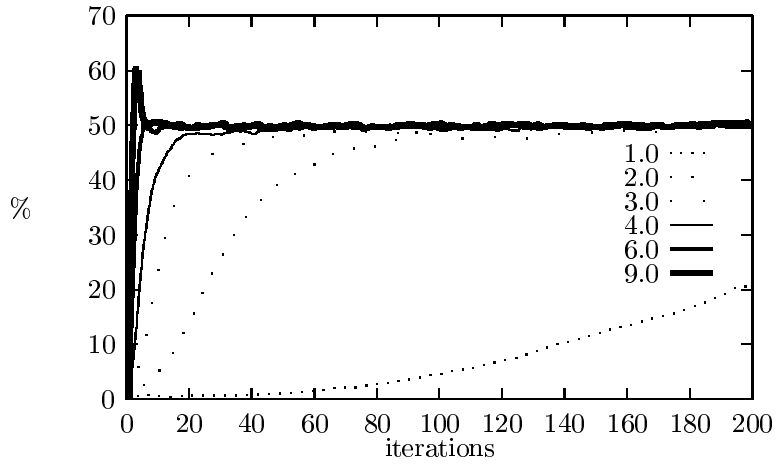

Figure 5.5. *Mean-convergence for Gibbs sampler. Percentage of 1-pixels versus the number of iterations for* $\beta = 3.0$.

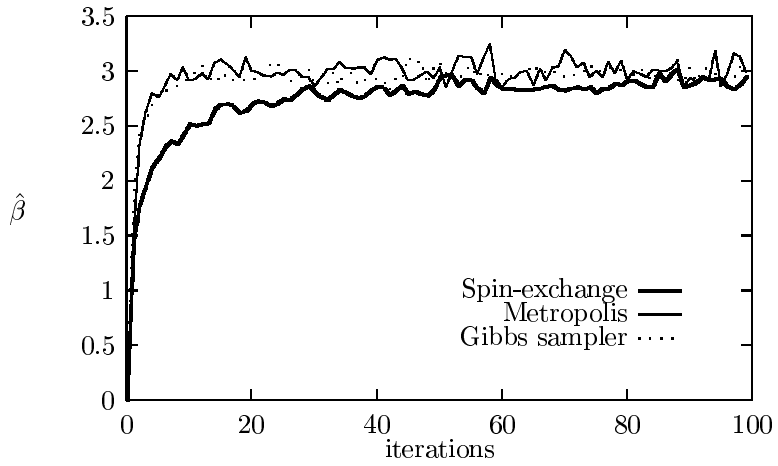

Figure 5.6. *Convergence in pseudo-likelihood* β *estimates for the Gibbs sampler, Metropolis spin-flip and Metropolis spin-exchange algorithms.*

For the spin-exchange, shown in figure 5.9, we have to wait 30000 iterations before the pattern stabilizes.

During these simulations we noticed that there were three different types of steady-state patterns. These are shown in figure 5.10. The first corresponds to the semi-steady state reported in Ripley & Kirkland (1990) for the unconditional simulation. The second pattern consists of one phase encapsulated in the other phase. The third pattern shows diagonal striping and this relies on the toroidal structure of the grid. Simulating 300 samples using 20000 iterations of the Metropolis spin-flip resulted in 56% of the first pattern, 41% of the second pattern and 6% of the third pattern. We then tried to simulate the same model with free boundary conditions, i.e. the pixels on the boundaries simply have less neighbors than nonboundary pixels. This

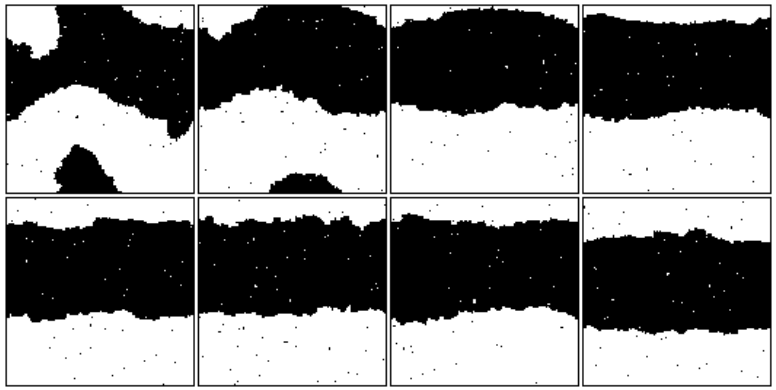

Figure 5.7. *Convergence of Metropolis spin-flip algorithm. Configurations after 50, 100, 200, 400, 800, 2000, 4000 and 8000 iterations for* $\beta = 3.0$.

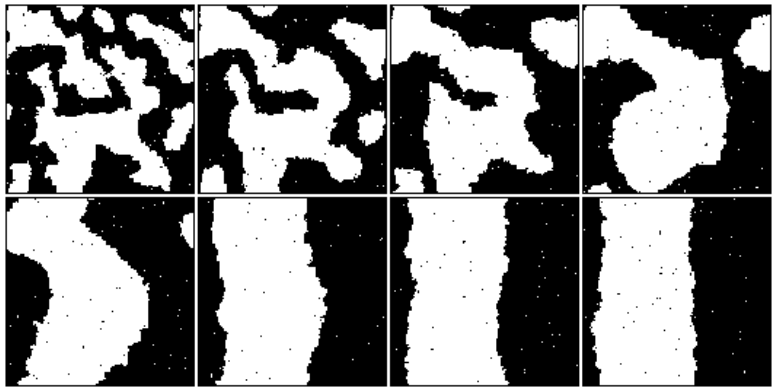

Figure 5.8. *Convergence of Gibbs sampler. Configurations after 50, 100, 200, 400, 800, 2000, 4000 and 8000 iterations for* $\beta = 3.0$.

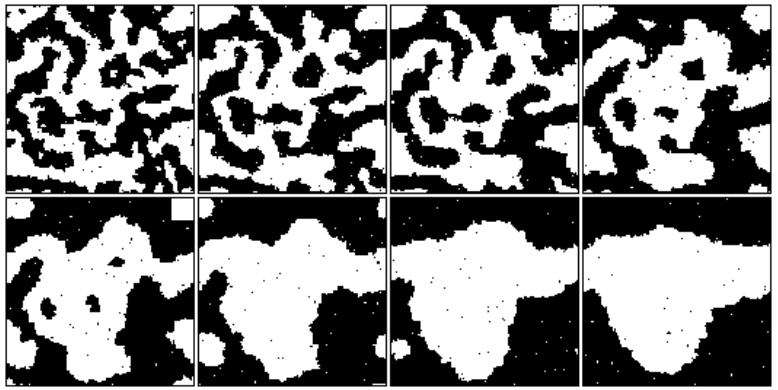

Figure 5.9. *Convergence of Metropolis spin-exchange algorithm. Configurations after 50, 100, 200, 400, 800, 2000, 4000 and 8000 iterations for* $\beta = 3.0$.

resulted in the two types of steady-state patterns shown in figure 5.11. At up to 100000 iterations the distribution was maintained at 80% of the left pattern and 20% of the right pattern.

These results also show that Cross & Jain (1983) never simulated to steady-state for supercritical β, and that the stop-criteria were more important in

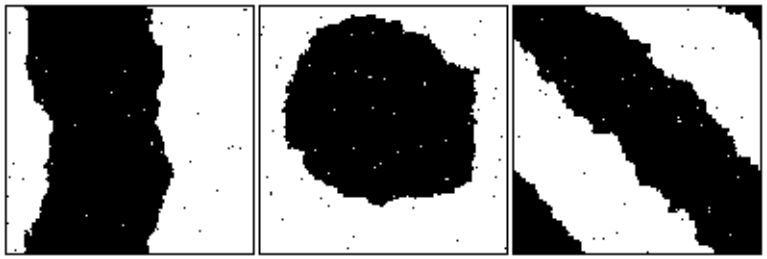

Figure 5.10. *Nature of steady-state patterns simulated on a toroidal grid.*

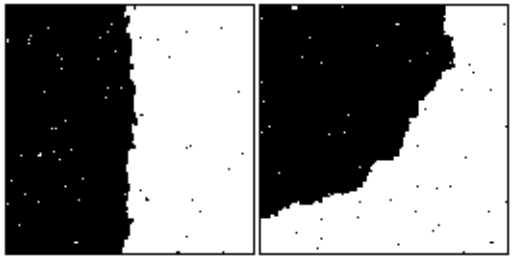

Figure 5.11. *Nature of steady-state patterns simulated with free boundary conditions.*

determining the visual properties of the simulated textures than the model parameters..

Timing

The parallel implementation of this algorithm on the CM-200 is in average 40 times faster than a sequential implementation on an HP Apollo 9000/750, which is marketed as the fastest workstation in the world at the moment. The time on the CM-200 was on an 8k processor system with exclusive access.

5.3.6 Conclusion

The iterative simulation of Markov random fields conditional on the first-order statistics has been studied. Until now such simulations has been done using the Metropolis spin-exchange algorithm, which was made popular by Cross & Jain (1983). Presented here are two spin-flip alternatives that have

several advantages. They are faster per sweep. The rate of convergence is higher, several orders of magnitude for supercritical β, and they are easy to parallelize. The essential part for the conditioning is a simple feedback loop. It is straightforward to extend the use of such a feedback loop in iterative simulation schemes to conditioning on other image features.

Using implementations of these algorithms on an SIMD massively parallel computer we have shown that Cross and Jain did not simulate to steady-state for large β and that their realizations for large β depends heavily on the stop-criteria used. Statistics has been made for the nature of steady-state configurations for both simulations on a toroidal grid and for simulations with free boundary conditions.

5.4 Simulation of geological structures

We use the morphological Potts models defined in equation (3.10) and apply the α-controlled spin-flip algorithm from section 5.3 for the simulation of geological structures in an oil field.

5.4.1 Introduction

In the business of petroleum exploration and production it is of great importance to assess the properties of an oil field. Computer simulation studies is a powerful tool in this assessment. They are performed by simulating flow in simulated stochastic reservoirs. Thus the word simulation is used in two senses in the field of petroleum technology. It is used for the stochastic simulation of the spatial distribution of sedimentary facies and petrophysical properties as well as for the numerical simulation of flow in a media. This case study is concerned with simulations in the first sense. When flow simulations is meant, this shall be stated explicitly.

The simulation is based on a reservoir model. In this model we have to incorporate geological knowledge from similar structures as well as the geological knowledge obtained from well data. The information used in the design of a reservoir model is often referred to as *soft data*. When simulating the reservoir model the well data shall be fixed at the corresponding location thus honoring what is called *hard data*. A source of information that seems somewhat harder to incorporate is data from seismic studies.

When modelling the distribution of rock types we use a discrete coding of the lithology, whereas models for petrophysical properties like porosity and permeability may be more naturally based on continuous variables.

For literature on the subject of this case study the reader is referred to Ripley (1992), Dubrule (1989) and Haldorsen, Brand, & Macdonald (1988).

5.4.2 Model types

The two main groups of stochastic models used in reservoir simulation are *object models* (or Boolean models) and *voxel models* (or block models). We are primarily concerned with voxel models.

Voxel models

These models are based on a regular grid and the distribution of voxel values is chosen to satisfy e.g. a certain variogram (correlogram) or a conditional probability distribution. The variogram and the conditional probability distribution may be inferred from hard data.

Three recent publications with different approaches are:

- Adler, Jacquin, & Quiblier (1990) simulated porous media based the measured porosity and variogram.
- Farmer (1989) generated gray level numerical rocks by first computing the histogram, cooccurrence matrices and autocorrelation of a rock

sample. Then a pattern with the same histogram is generated, and this pattern is used as a starting configuration for a spin-exchange simulated annealing procedure. Deviation from the sample cooccurrence matrices and autocorrelation is used as penalty in the energy function.

- Ripley (1992) simulates the distribution of rock types using a 3D Potts model conditional on the hard data points.

The approach in this study is similar to the approach in Ripley (1992). We include morphological properties in the model by using the morphological Potts models defined in equation (3.10), and we apply the α-controlled spin-flip algorithm from section 5.3 for the simulation.

Object models

An object model describes the distribution of rock bodies of random shape at random locations. The theory of point processes and random set models can be a very useful tool for specifying and simulating object models. From the viewpoint of texture analysis object models corresponds to the structural approach with primitives and placement rules.

5.4.3 A Markov random field reservoir model

We shall try to apply the morphological Potts model 3.10 for reservoir simulation on the gigascopic scale (Haldorsen et al., 1988). The model is intended for the description of the distribution of both rock types and discretized

petrophysical properties. The goal is that the reservoir simulation scheme shall incorporate:

- Fixed first-order statistics.
- Anisotropy in the different facies.
- Spatial trends (instationarity).
- Hard data.
- Planar discontinuities (faults) as hard data.

The simulation is performed using the feedback loop described in section 5.3 to keep the first-order statistics fixed. The constants for the control actions has to be selected for each state. A PI-controller was used in the simulations below.

Anisotropy in the different facies is implemented through the use of the morphological Potts models.

Instationarity can be implemented by letting the model parameters vary across the field.

Hard data points are honored by simply not visiting them during the simulation, i.e. they will never change their value.

Discontinuities (faults) are introduced as hard data. This is simply done by considering discontinuities as a new phase, the *discontinuity phase* (The discontinuity phase can also be considered as vacancies in a site-diluted Potts model). The discontinuity phase is not considered in the flipping

process. To be 100% effective the discontinuity phase has to be as wide as the longest distance between two neighbors in the MRF. To avoid the wrap-around that is due to the toroidal grid we can apply the discontinuity phase to the sides of the grid. Horizontal wrap-around may be desirable in many cases, whereas this is rarely true for vertical wrap-around. We would in this case apply the discontinuity phase to the top and/or bottom lines of the grid. An alternative to the discontinuity phase is to use a bond-diluted Potts model. In such a model we have no bonds (no interactions) across the discontinuity zone.

5.4.4 Simulation results

A reservoir simulation program, *rocksample*, has been implemented (See appendix A). We shall now show a few examples of simulations in 2D based on these models. The simulations were made on a 128x128 grid, where the pixels are rectangles with height 1 and length 4. The model used is a four-state morphological Potts model with the two structuring elements shown in figure 3.13. In figures 5.12 to 5.15 we show four simulation examples. 50 iterations of the Metropolis spin-flip algorithm were used. Simulations like these confirmed that first-order statistics and anisotropy of the different facies can be controlled.

Figures 5.16 and 5.17 illustrates the conditioning on hard data and faults as hard data. The simulations are conditional on hard data in vertical columns on both sides of the fault. Figure 5.16 shows two independent simulations, one with a fault and one without. In figure 5.17 the fault is introduced in the result of the simulation without the fault and then simulation is done again. In this case the simulations look more similar.

Figure 5.12. Simulation result of four-state morphological Potts model.

Figure 5.13. Simulation result of four-state morphological Potts model.

Figure 5.14. Simulation result of four-state morphological Potts model.

Figure 5.15. Simulation result of four-state morphological Potts model.

Instationarity has not been implemented in the simulation program.

All the parameters used in the simulations were supercritical, i.e. the steady-state result would have only one color if we did not condition on the first-order statistics. Thus for these simulations to be useful in practice we have to find a suitable stop-criterion. The stop-criterion could be a global structural statistic, e.g. average cluster size or length.

5.4.5 Conclusion

The usefulness of Markov random fields in reservoir simulation is dependent on an efficient implementation of the simulation scheme. This is particularly true when simulating 3D structures. The α-controlled spin-flip algorithm implemented on a massively parallel computer has proved very efficient and would be an appropriate choice. The examples presented in this section has been simulated on a serial workstation in 2D, but they are easily extended to 3D.

Figure 5.16. Result of two simulations with identical parameters. The lower image has a fault as hard data. The simulations are conditional on hard data in vertical columns on both sides of the fault.

Figure 5.17. Result of two simulations with identical parameters. The lower image is made by introducing a fault in the upper image and then simulating again. The simulations are conditional on hard data in vertical columns on both sides of the fault.

Simulations based on the morphological Potts model suggest that it is possible to satisfy a set of criteria that are relevant to reservoir geologists.

The number of simulations that we have computed so far is very limited, and much more research is needed to evaluate the potential of morhological Markov random fields (and Markov random fields in general) in reservoir simulation.

Chapter 6

Bayesian paradigm

The *Bayesian paradigm* is a framework for incorporating stochastic models of visual phenomena into a very general set of tasks from image processing and image analysis. Since the seminal paper of Geman & Geman (1984) there has been an increasing interest in this subject (Besag, 1986; Marroquin, Miter, & Poggio, 1987; Geman & McClure, 1987; Ripley, 1988; Besag, 1989; Geman, Geman, Graffigne, & Dong, 1990). We give a short review of Bayesian image analysis and present an application that makes successful use of Markov random fields, the Metropolis algorithm and simulated annealing in a Bayesian framework.

6.1 Introduction

The Bayesian paradigm in image analysis can be described as follows:

1. We construct a prior probability distribution P($\mathbf{x}$) for the visual phenomena $\mathbf{X}$, that we want to make inferences about.
2. We then formulate an *observation model* $P(\mathbf{y} \mid \mathbf{x})$. This is the distribution of observed images $\mathbf{Y}$ given any particular realization $\mathbf{x}$ of the prior distribution.
3. The prior distribution and the observation model are combined to the posterior distribution $P(\mathbf{x} \mid \mathbf{y})$ by Bayes theorem

$$P(\mathbf{x} \mid \mathbf{y}) \propto P(\mathbf{y} \mid \mathbf{x})P(\mathbf{x}).$$

   $P(\mathbf{x} \mid \mathbf{y})$ is the distribution of the visual phenomena $\mathbf{X}$ given the image $\mathbf{y}$ that we have observed.
4. Finally we make inferences about the visual phenomena based on the posterior distribution $P(\mathbf{x} \mid \mathbf{y})$.

6.2 Prior distribution

The generality of Bayesian image analysis lies in the variety of visual phenomena, that we can model.

In image restoration we want to make inferences about the true undegraded image represented by $\mathbf{X}$ from a noisy observed image $\mathbf{y}$. A Gaussian MRF

could the be an appropriate prior distribution for $\mathbf{X}$. Priors that model the joint gray level distribution are called *pixel priors*.

The goal of image classification is to assign a class or label to each pixel in an image $\mathbf{y}$. E.g. in remote sensing we can assign land-use classes like forest, lake, road etc. to pixels in satellite images. The joint assignment of labels to all pixels is a *labelling* $\mathbf{x}$. Priors P($\mathbf{x}$) for the labelling could be discrete Markov random fields such as binary MRFs and Potts models. Priors that model a labelling are called *label priors*.

If we want to make inferences about geometrical shapes, represented by $\mathbf{X}$, in an image $\mathbf{y}$, we are in the area of template matching. *Template priors* are models of geometrical relations in objects or between objects in an image. The application of a MRF template prior is illustrated in the case study of section 6.6.

For the Bayesian approach to be successful it is important that the prior density reflects our knowledge of the visual phenomena behind the observed images.

6.3 Observation model

The observation model $P(\mathbf{y} \mid \mathbf{x})$ is the distribution of observed images $\mathbf{Y}$ given any particular realization $\mathbf{x}$ of the prior distribution, i.e. it tells us how the visual phenomena, that we want to make inferences about, is actually observed. In image restoration $\mathbf{x}$ is typically considered observed after convolution with a blurring function h and addition of a noise image

ϵ, as

$$\mathbf{Y} = h * \mathbf{x} + \epsilon.$$

In image classification the observation model could be a texture and/or noise model for each class, e.g. a forest texture on forest labels, a lake texture on lake labels etc.

After having specified the prior model and the observation model we are ready to extract information from the posterior distribution.

6.4 Maximum a posteriori (MAP) estimates

The MAP estimate $\hat{}\mathbf{x}$ of $\mathbf{x}$ given an observed image $\mathbf{y}$ is defined by

$$\hat{\mathbf{x}} = \arg\max P(\mathbf{x} \mid \mathbf{y}).$$

Thus MAP estimation involves maximization of a high dimensional joint distribution, and this is usually connected with a considerable computational cost.

6.4.1 Simulated annealing

A simulated annealing scheme is a successive sampling from the density

$$P_T(\mathbf{x} \mid \mathbf{y}) \propto [P(\mathbf{y} \mid \mathbf{x})P(\mathbf{x})]^{\frac{1}{T}} \tag{6.1}$$

where the temperature starts at an initial value $T_0 > 0$ and then falls towards 0. If the temperature is lowered slow enough, then (6.1) will assign

unit probability to the MAP image in the limit (Geman & Geman, 1984). The sampling algorithm for simulated annealing can be e.g. the Gibbs sampler or the Metropolis spin-flip algorithm. A simulated annealing scheme will be used in section 6.6. The reader is referred to Aarts & Korst (1989) for details on the simulated annealing algorithm.

6.4.2 Iterated conditional modes (ICM)

The ICM algorithm consists of a number of sweeps over the image, where each pixel is visited and set to the mode of the conditional probability, i.e.

$$\hat{x_i} = \arg\max_t P(Y_i \mid y_i) P(X_i = t \mid x_j, j \neq i).$$

The ICM algorithm usually converges in the order of 10 sweeps, which is generally much less than would be required for a simulated annealing scheme. An ICM scheme is on the other hand more likely to get trapped in a local maximum of the posterior density.

6.5 Marginal posterior modes (MPM)

Marroquin et al. (1987) generated a series of samples from the (discrete) posterior distribution and, for each pixel, chose the mode of the marginal posterior distribution, i.e.

$$x_i^* = \arg\max P(x_i \mid \mathbf{y})$$

The sampling algorithm for MPM can be e.g. the Gibbs sampler or the Metropolis spin-flip algorithm. A MPM scheme used with a label prior will

minimize the expected misclassification error under the posterior distribution.

6.6 Hybridization filter analysis

An algorithm for automatic localization and classification of spots on a hybridization filter has been developed and implemented. The algorithm represents a successful application of a Markov template prior, the Metropolis algorithm and a simulated annealing scheme.

6.6.1 Background

The genome analysis lab at Imperial Cancer Research Fund (ICRF) in London is working on the human genome project. This project involves a massive amount of hybridization experiments. The intention of the work presented here is to analyze hybridization filters automatically for the mapping of the human genome.

The filter is a square sheet of nylon with a sidelength of 23.2 cm. A robot places a 96x96 grid of spots on this filter, where each spot is a specific cosmid clone. A cosmid clone is a stretch of DNA, about 40000 bases long. When a radioactive DNA probe is applied to the filter the probe will only bind (hybridize) to those cosmid clones that contain the same DNA sequence as the probe itself. The unbound probes are washed off, and spots containing clones hybridized to the probe appear darker than the other spots, when an *autoradiograph* is taken of the filter. When a *phosphor image* is taken the spots containing hybridized clones will appear lighter than the other spots.

6.6.2 Robot dynamics

The cosmid clones are placed on the filter by a robot. They are kept on microtiter dishes with an 8x12 grid of wells, thus the robot arm consists of an array of 8x12 pins. When the robot arm is dipped into a microtiter dish a small quantity of each cosmid clone adheres to its corresponding pin. The arm is then moved to the filter where it applies the cosmid clones as an array of spots. After this the robot sterilizes the pins and moves on to the next dish. This is done 96 times for each filter producing the 96x96 grid of spots. This grid is made up of 6 almost independent 32x48 subgrids as shown in figure 6.1. Each subgrid contains 4x4 interleaved 8x12 grids corresponding to the microtiter dish grid. The spacing between the wells of the microtiter dishes is 8 mm, thus the spacing between the spots is 2 mm.

6.6.3 Image analysis problem

The problem to be solved through the use of image analysis is to automatically detect which cosmid clones hybridizes to the probe. This involves the correct assignment of each spot on the filter to a corresponding region in the image and classification of each spot as to the degree to which hybridization has occurred. Several circumstances complicate the solution of these problems. For the spot localization problem we have that

- The robot movements are imprecise.
- The membrane may physically warp.
- Some pins of the robot arm may be bent.

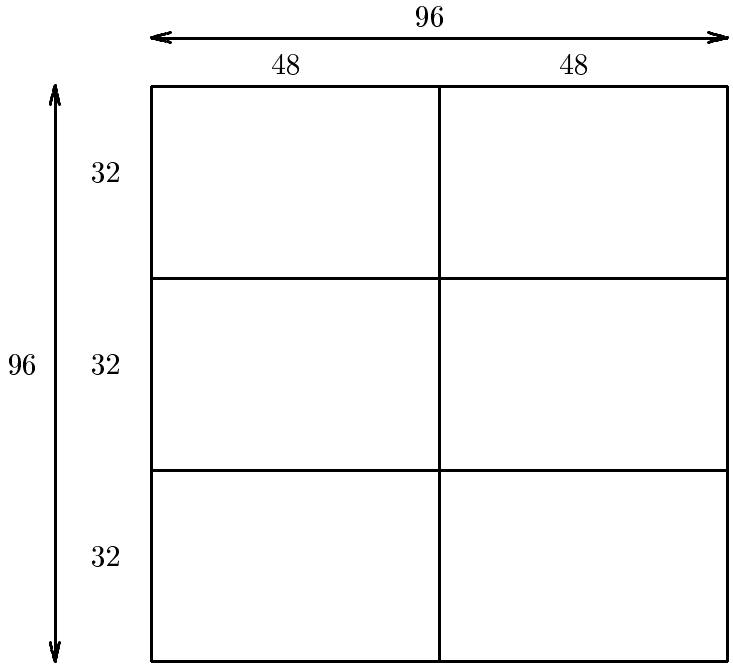

Figure 6.1. Arrangement of the 6 subgrids. The full grid is a 96x96 spot array.

- Some spots are missing.
- Some spots may have merged.

For the spot classification problem we have that

- The background radiation level varies across the filter.
- Some spots may have merged.
- Some spots may have been misplaced.

We attempt to provide an effective set of image analysis tools that are robust under these circumstances. The spot localization problem is considered to be the most difficult and will be our main concern.

6.6.4 Digitization

The autoradiograph is digitized by a camera and a framegrabber. For the setup used now the result is an 8 bit 512x512 image. The gray-scale of these images is inverted to get white spots.

The phosphor imager scans with 88 μm per pixel. It is capable of scanning an area of up to 35 $\times$ 43cm with 16-bit gray-scale resolution. The filter and its immediate surroundings are scanned and the resulting image is sub-sampled to an 8-bit 1024x1024 image. This image is the starting point of the processing. The spacing between the spots is about 8.5 pixels and this seems reasonable for our purpose. The examples shown in this thesis are phosphor images. In figure 6.2 we show an example of a raw image.

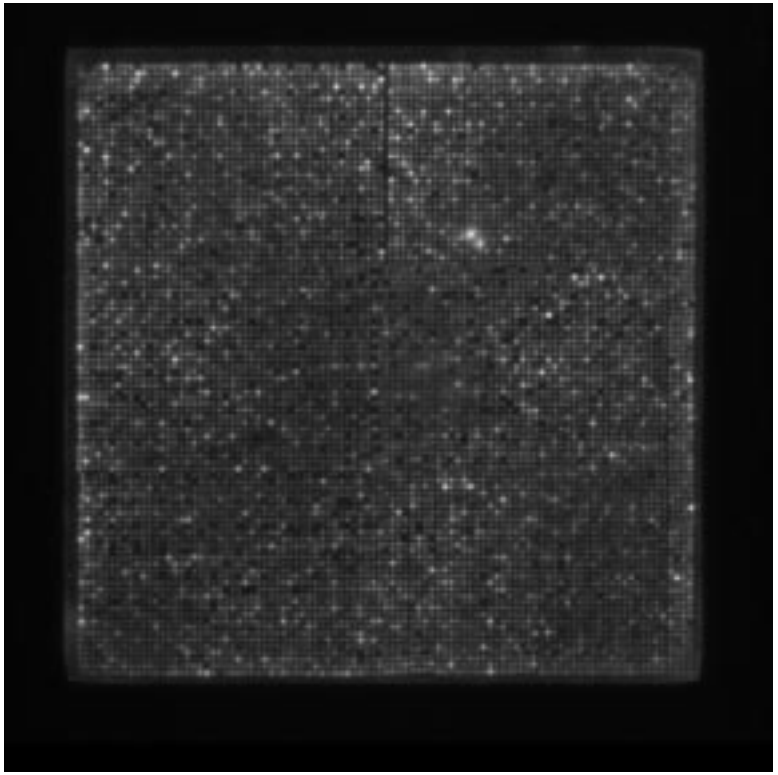

Figure 6.2. Raw image. This is a good quality phosphor image showing the full 96x96 spot array.

6.6.5 Preprocessing

The preprocessing serves four purposes.

1. Correction for rotation.
2. Finding the rectangular outline of the spot array.
3. Correction for background variations.
4. Spot equalization.

The success of the subsequent spot localization and spot classification depends highly on a successful implementation of these preprocessing steps.

To illustrate the preprocessing, the spot localization, and the spot classification we will show the effect of each step on the image in figure 6.3. This is a phosphor image of a 32x48 subgrid.

Correction for rotation

The spot array is normally very well aligned with the pixel array in phosphor images, but autoradiographs will in general be rotated slightly. The rotation angle can be found by using the Hough transform (See e.g. Duda & Hart (1972)) and search for the angle between e.g. -5 and +5 degrees with the highest variance over the profile in Hough space. The image can then be rotated back into alignment. In figure 6.4 we see the image from figure 6.3 after alignment.

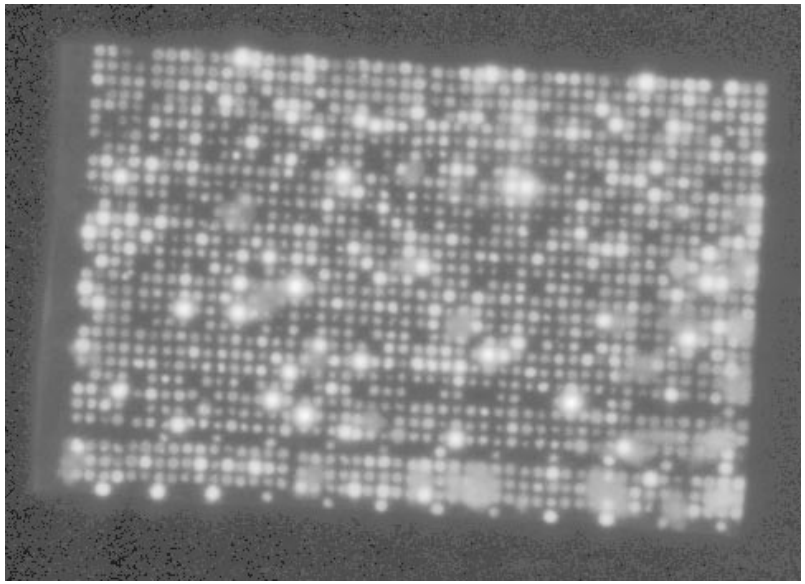

Figure 6.3. Phosphor image of a 32x48 subgrid.

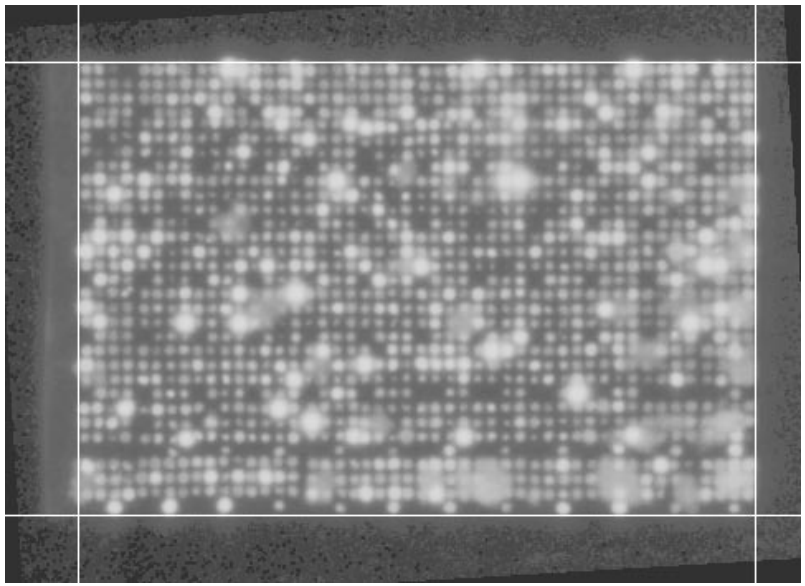

Figure 6.4. Aligned version of the image in figure 6.3. The rectangular outline is shown.

In particularly hard cases the four corners of the spot array can be pointed out manually. In this way we can align the image and obtain the rectangular outline of the spot array.

Finding the rectangular outline of the spot array

We can now assume strictly horizontal and vertical borders on the spot array. These borders are found by first computing the sum of each row (column), $\{s_i, i = 0, .., 1023\}$. Then we compute the difference of lag 8 obtaining $\{d_i, i = 8, .., 1023\}$, where $d_i = s_i - s_{i-8}$. Lag 8 is chosen because it is close to the distance between spot rows (columns). Finally we find the starting row (column) and ending row (column) as $\arg\max_i d_i - 4$ and $\arg\min_i d_i + 4$, where $\arg\max_i d_i$ is the row (column) number with the maximum difference, and $\arg\min_i d_i$ is the row (column) number with the minimum difference. Figure 6.4 shows the rectangular outline, where three of the sides were found by this method. The ending row had to be repositioned.

Correction for background variations

The background varies over the images and this will cause problems in the localization and classification process. The standard way of correcting for varying background is to subtract a lowpass filtered image from the original. As a lowpass filter we will choose a gray-scale opening. Using a flat 9×9 gray-scale opening will remove all the spots and leave the background, which is then subtracted from the image. This operation can be written as

$$R = I - O(I)$$

where I is the input image, O(I) is the opening of the input image, and R is the resulting image.

Spot equalization

To make the localization process easier we equalize the intensity of the spots, thus weighting the spots equally. This is done using a *morphological equalization*,

$$R = \frac{I}{D(I) - E(I)}$$

where I is the input image, D is the dilated input image, E is the eroded input image, and R is the resulting image. Again a flat 9×9 structuring element is used. Basically the morphological equalization makes the local gray level range constant over the image. Figure 6.5 shows the effect of background correction and spot equalization of the image in figure 6.4.

6.6.6 Spot localization

The spot localization involves matching a 96×96 grid on the spots in the image. This grid should adapt globally to provide the absolute spot coordinates and locally to take into account all the small distortions in the grid.

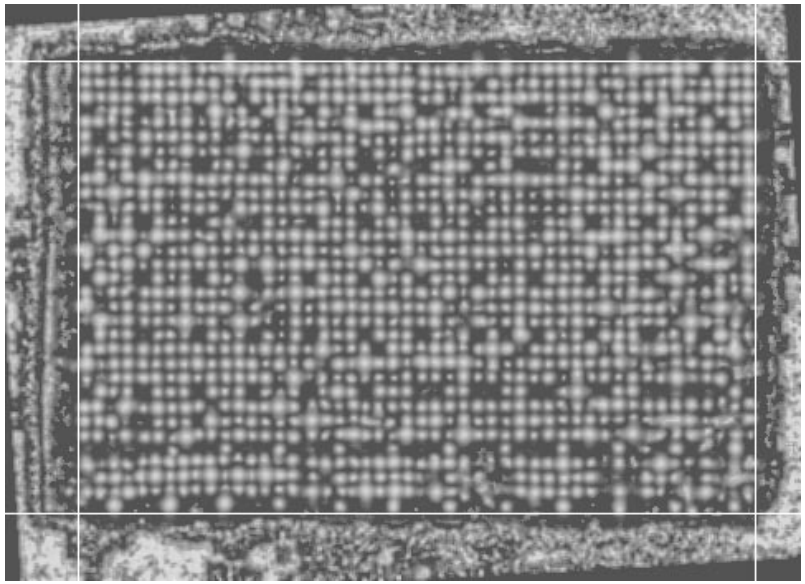

Figure 6.5. Background corrected and spot equalized version of the image in figure 6.4.

Initial assignment

If we can obtain a good initial guess on the spot locations then the subsequent processing will be faster. As the initial guess we cover the outline rectangle with a regular 96×96 grid.

Simulated annealing scheme

As a basis for improving the spot locations we use a Markov random field as a template prior for the regular grid structure of the spot array. The variables in this model does not represent a pixel value but the (x, y) image position of a spot. This variable is not defined on the pixel grid but on the spot grid. The prior is defined as

$$P(\mathbf{g}) \propto \exp(-\beta_0 \sum_{i \sim j} d_0^2(i,j) - \beta_1 \sum_{i \sim j} (d_1(i,j) - D)^2)$$

where i and j represents spots and $\mathbf{g} = \{(x_i, y_i), i = 1, .., n_s\}$ contains the locations (x, y) of all the spots. n_s is the number of spots in the spot array. The neighborhood is the four nearest neighbors. $d_0(i,j)$ is the deviation in alignment of the spots i and j, and $d_1(i,j) - D$ is the deviation from the fixed grid distance, D, between neighbors. Figure 6.6 illustrates the meaning of d_0 and d_1.

Given the spot locations $\mathbf{g}$ we then specify an observation model for the observed image $\mathbf{y}$ as

$$P(\mathbf{y}|\mathbf{g}) \propto \exp(\alpha \sum_i \mu(i))$$

where the summation is over all spots i, and $\mu(i)$ is the sum of the gray levels in a 5x5 neighborhood around spot i.

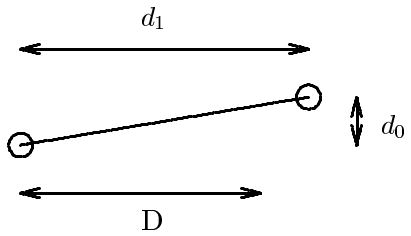

Figure 6.6. Definition of the distance measures d_0 and d_1 for a horizontal neighbor-pair. D is the distance between neighbors on a perfect grid.

We can regard this setup as a structural texture model. The prior model represents the placement rules and the observation model represents the primitives.

The posterior distribution

$$P(\mathbf{g}|\mathbf{y}) \propto P(\mathbf{y}|\mathbf{g})P(\mathbf{g})$$

is obviously a new Markov random field, and reflects a trade-off between the regularity of the grid and the trust in the image data. The energy function of the posterior distribution is given as

$$U = \beta_0 \sum_{i\sim j} d_0^2(i,j) + \beta_1 \sum_{i\sim j} (d_1(i,j) - D)^2 - \alpha \sum_i \mu(i).$$

In this energy function we can control the properties of the fitted grid. The faith in the data is controlled by α, since this parameter is the weight of the intensity of the spots. The regularity of the grid is controlled by β_0 and β_1. β_0 determines the degree of linearity of the grid and β_1 controls deviations from the fixed grid distance between neighboring spots.

This model has a problem for spots on the edges. If nothing is done the three or four outer rows and columns will be dragged towards the center of the spot array because of the lack of spots pulling the other way. To eliminate this effect we define artificial spots around the edges of the spot array. The artificial spots are initially positioned just outside the outline of the spot array, and the only restriction in their movements is that they cannot cross this outline.

We can now apply a simulated annealing scheme and the Metropolis algorithm using this Markov random field. Every spot is visited and an attempt is made to change its position to a randomly selected nearest neighbor.

In figure 6.7 the location of every spot in figure 6.5 is marked by a dot. Figure 6.8 show the spot locations in a close-up of the lower right corner of figure 6.7.

Robot grid control

The same 8×12 grid of robot pins is used 96 times on each filter. This grid can be regarded as fixed and we can use that information to detect and correct misplaced spots. We first compute the mean of the relative positions of neighbors in the 8×12 grid. Then for each spot we compute the deviations from this mean for all four neighbors. The trimmed mean (min and max trimmed off) of these four deviations will give a grid deviation number for each spot. If the grid deviation number exceeds a specific value the spot will be considered as misplaced. The misplaced spots can then be relocated using the relative position to the neighbor in the 8×12 grid with the lowest grid deviation.

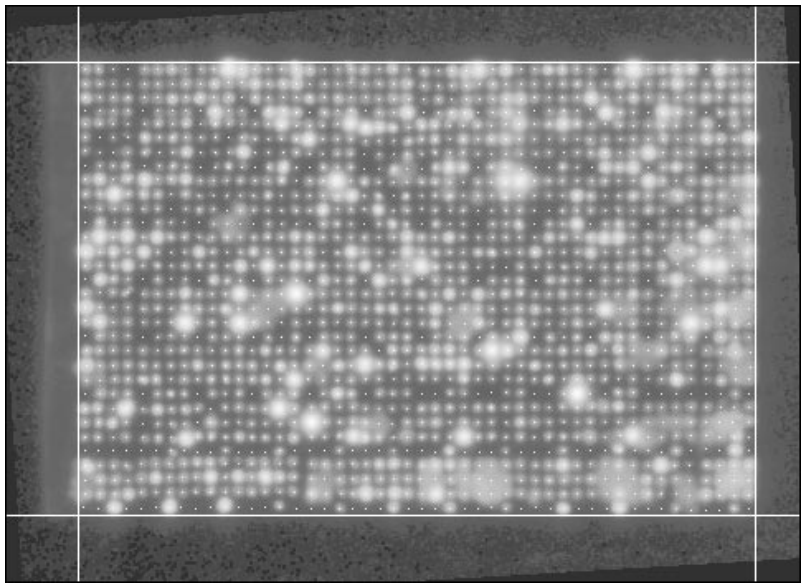

Figure 6.7. Spot locations. The located spots of the image in figure 6.5 are marked with a dot.

Figure 6.8. Spot locations in a close-up of the lower right corner of the image in figure 6.7.

There are no misplaced spots in figure 6.7, but an example of the robot grid control procedure will be shown in section 6.6.8.

6.6.7 Spot classification

The spot classification is based on the mean gray level in the neighborhood of the spot location from the background corrected image. Thresholds are selected to classify each spot in one of three classes: positive (+), negative (-) or missing (x). Spot classifications of the located spots in figure 6.8 are shown in figure 6.9.

If there is any doubt whether a spot has been correctly located it will be classified as missing.

6.6.8 Results

Figures 6.10 and 6.11 show close-ups of the localization result of the image in figure 6.2. They illustrate the robustness of the algorithm. In figure 6.10 there is a vertical gap down the middle of the image. This gap does not cause any problems in the localizations. In the center of figure 6.11 we see that two rows of spots merge and split up again. This is also interpreted correctly by the algorithm. In both figures we see that missing spots are located in a satisfactory way. To obtain these results we used the parameters:

- Prior model: $\beta_0 = \beta_1 = 5.0$
- Observation model: $\alpha = 0.2$

Figure 6.9. Spot classifications of the located spots in figure 6.8. The classes are: positive (+), negative (-) or missing (x).

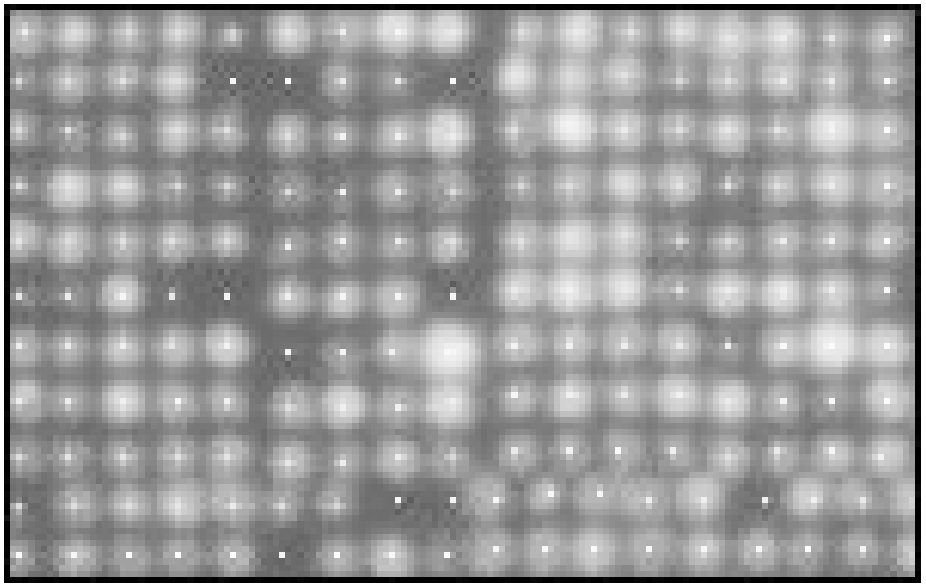

Figure 6.10. Close-ups of the localization result of the image in figure 6.2. There is a vertical gap down the middle of the image.

- Starting temperature: $T_0 = 4.0$
- Temperature scheme: $T_n = T_{n-1}\frac{\log(n+2)}{\log(n+3)}$
- Number of iterations: 100

Until now we have only shown good quality phosphor images. In figure 6.12 we see a noisy phosphor image, where the regular spot pattern is hardly noticeable in large areas of the spot array. In the simulated annealing scheme we used the same parameters as before except that we set $\alpha = 0.1$ to put less trust in the data and more trust in the grid structure. The locations found on the full grid are marked with dots and shown in figure 6.13. A close-up of this figure is shown in figure 6.14. We can see from the figures, that the

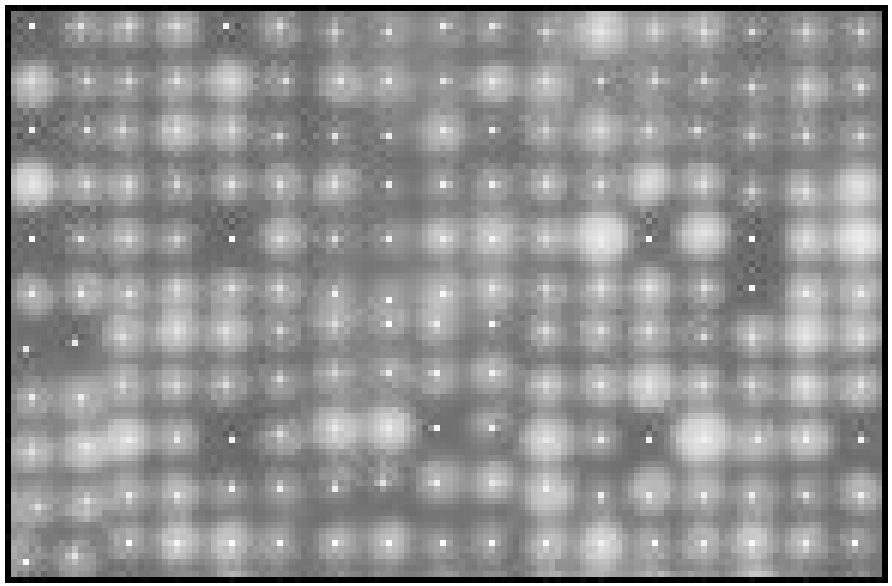

Figure 6.11. Close-ups of the localization result of the image in figure 6.2. In the center we see that two rows of spots merge and split up again.

Figure 6.12. Noisy phosphor image. The regular spot pattern is hardly noticeable in large areas of the spot array.

Figure 6.13. Spot locations from the image in figure 6.12 marked with dots.

clearly visible spots are located correctly. Even for areas where spots are hardly noticeable we see that algorithm makes a reasonable choice.

Figures 6.15 and 6.16 show the effect of the robot grid control. A group of 4x3 spots has been shifted to the left in figure 6.15. In this case the shift was due to a fast cooling. In figure 6.16 we see the misplaced spots pointed out by the robot grid control algorithm. We can now relocate the misplaced spots and run the localization algorithm again.

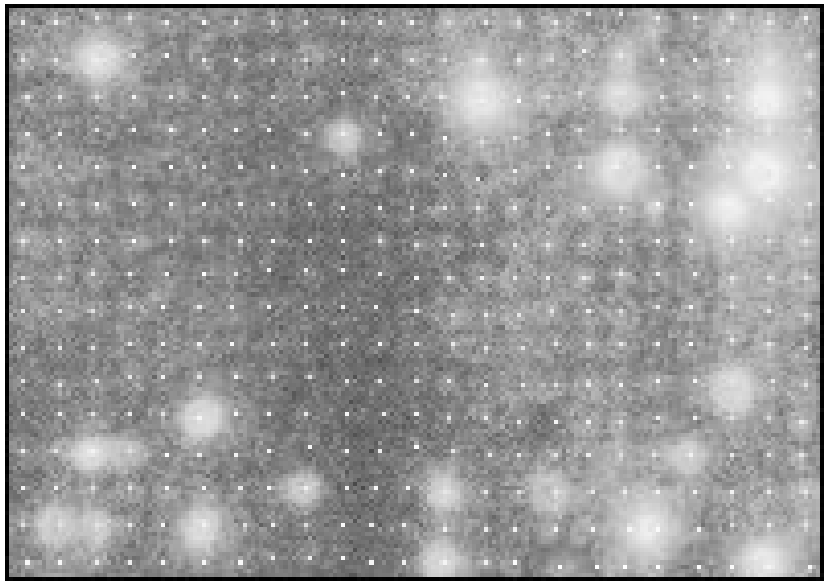

Figure 6.14. Close-up of the spot locations shown in figure 6.13.

Figure 6.15. Errors in the localization. A group of 4x3 spots has been shifted to the left.

Figure 6.16. Misplaced spots pointed out by the robot grid control algorithm.

6.6.9 Conclusion

We have presented an algorithm for automatic localization and classification of spots on a hybridization filter. The algorithm is based on a Markov template prior for the spot array, and the localization is obtained as a trade-off between this model and the observed data. The computation is based on a simulated annealing scheme. A set of operations was used to preprocess the images. These preprocessing steps helped significantly in making the simulated annealing scheme successful and computationally feasible. A postprocessing step that implements a check on the localization has been implemented.

The algorithm has been successfully applied to many hybridization filter images. It seems to be both effective and robust compared to previously tested automatic methods (unpublished). It seems to be able to classify spots much faster and in many cases more accurately than a manual operator.

Chapter 7

Conclusion

Texture is an important characteristic of visual phenomena, and many attempts have been made to capture the relevant textural properties in a set of texture features or as a texture model. We have contributed to these attempts by going through selected theory and practical applications.

7.1 Summary

For texture description we have based our studies on the first- and second-order statistics. We have shown that first-order statistics can provide valuable textural information if they are computed at several scales (resolutions). We found that a coarse-scale first-order statistic robustly measured enzymatic treatment effects on textile. This shows that it may be fruitful to

consider the simplest features first, when solving a texture description problem.

We have surveyed features based on gray level cooccurrence matrices. The effect of matching the gray level histogram to a specific distribution before computing the cooccurrence features has been studied. Classification results suggest that the frequently used histogram equalization reduces the discriminatory power of the features significantly for stochastic textures. A relatively neglected feature, the diagonal moment, turned out to be very important for discriminating textures after a Gaussian histogram match. This suggests that in general we loose important information when replacing the cooccurrence matrix with the gray level difference and gray level sum histograms. The combination of Gaussian matched textures and CART classification resulted in simple, easily interpretable and relatively accurate classifiers.

Markov random fields have been surveyed as texture models. Many important results about these models from the field of statistical physics are still fairly unknown in the field of image analysis. We have restated some of the results in a statistical setting. These results leads us to an extension to the asymptotic maximum likelihood estimator of Pickard (1987).

Standard Markov random fields are based on pairwise interaction between pixels thus failing to incorporate morphological properties. We suggest a reformulation of the discrete models, in which the operators of mathematical morphology replace the concept of cliques. The advantages of morphological Markov random fields are, that morphological properties become more apparent and that we obtain a coherence between texture description and

texture models. Illustrative simulations of morphological Markov random fields show that interesting visual phenomena can be created.

We have given a review of Markov random field parameter estimation and Markov random field simulation. A new, fast, parallel algorithm for simulation conditional on the first-order statistics has been developed and implemented on a massively parallel computer. The conditioning is maintained by a standard PID-controller. Long runs of this algorithm has given us information about steady-state patterns for the conditional models. The algorithm has also been used for simulations of the geometrical structure of oil reservoirs based on a morphological Markov random field model.

Markov random fields have been used successfully in a Bayesian setting to analyze hybridization filters automatically for the human genome project. A first-order Markov random field is used to model the geometrical structure of a spot array, and this model is then used as prior knowledge for the accurate localization of the single spots. The localization is done using a simulated annealing scheme.

An extensive collection of software has been developed during the course of this work. The main software developments are listed in appendix A.

7.2 A comment

Texture analysis has been studied extensively by many researchers over the last two decades. The standard reference for most of these studies has been the Brodatz textures. Although these textures can continue to provide

insight about texture features, there are two points of criticism to such an approach. The Brodatz textures only represent an infinitesimal fraction of real world textures, and the Brodatz textures are very different. Even though the literature on texture analysis, based on Brodatz textures, is full of successes, there are still plenty of challenges for texture researchers in fields like industrial inspection, biological and medical imaging, remote sensing, geology etc.

Appendix A

Developed software

An extensive selection of software has been developed during the course of this work. The serial (nonparallel) programs were developed in C on HP workstations running HP-UX. The parallel programs were developed in C^* on a Connection Machine CM-200 with a Sun-4 front end. Serial programs have the suffix **.c** and parallel programs have the suffix **.cs**.

Standard numerical algorithms were taken from Press et al. (1988). On the Connection Machine we used the supplied CMSSL library. For random number generation under HP-UX we used the well-known linear congruential algorithm with 48-bit integer arithmetic (drand48). On the Connection Machine we used a lagged-Fibonacci algorithm (Knuth, 1973) implemented in the CMSSL library.

Both serial and parallel programs have been made to work as modules of the pipe-oriented HIPS and HIPS-2 image processing software.

The following list contains the main software developments.

- Texture statistics

  1. **histinfo**
     Histinfo computes first-order statistics from the input image.
  2. **glcm**
     Glcm computes the gray level cooccurrence matrix and 15 features from this matrix.
  3. **fhist**
     Fhist takes a floating point input image, sorts all the pixels, and outputs byte image with a specified histogram. The histogram can be uniform (equalization), Gaussian, or a beta-function.

- Markov random field estimation

  1. **binest**
     Binest computes coding estimates and maximum pseudolikelihood estimates from a binary input image. Models up to order five can be estimated. Isotropy/anisotropy can be controlled for each neighbor-distance. The χ^2 test statistic and the log-likelihood is computed for each coding.
  2. **binomest**
     Binomest estimates the maximum pseudolikelihood estimates of a binomial Markov random field from the input image.

3. **asympest**
   Asympest computes the asymptotic maximum likelihood estimate of a first-order binary Markov random field from the input image.

4. **pottsest**
   Pottsest computes the maximum pseudolikelihood estimate of a Potts model from the input image.

5. **gaussest**
   Gaussest computes the maximum pseudolikelihood estimate of a Gaussian Markov random field from the input image.

- Markov random field simulation

1. **binsamp**
   Binsamp simulates binary Markov random fields using the Gibbs sampler or the Metropolis algorithm.

2. **pottssamp**
   Pottssamp simulates Potts models using the Gibbs sampler or the Metropolis algorithm.

3. **morphsamp**
   Morphsamp simulates morphological binary Markov random fields using the Metropolis algorithm.

4. **swendsen**
   Swendsen simulates Potts models using the Swendsen-Wang algorithm.

5. **rocksamp**
   Rocksamp simulates geological samples using a morphological Potts model and the α-controlled spin-flip algorithm. Model parameters for each phase can be specified. Rocksamp is the program used in section 5.4.

6. **icrf**
   Icrf is the package of hybridization analysis software used in section 6.6.

7. **bingen**
   Bingen is program in C^* for Connection Machines. It simulates binary Markov random fields in 2D and 3D. The α-controlled algorithm is implemented. The results can be monitored "real time" in an X-window.

- Other

1. **xshow**
   Xshow is a program that displays HIPS images under X-windows and lets the user interact using HIPS programs.

2. **frarithmetic**
   Frarithmetic is a program that can be executed with many names (all starting with "fr"). It does many kinds of arithmetic operations on a set of images.

3. A HIPS implementation of the basic gray level morphological operations:
   - Erosion
   - Dilation
   - Opening
   - Closing
   - Morphological gradient
   - White tophat

- Black tophat
- Morphological equalization

Appendix B

GLCM for all Brodatz textures

This appendix contains the right-neighbor GLCM for all the Brodatz textures (no histogram match). By observing the kind of structures these matrices can have we may get a better idea of which features give the best summary.

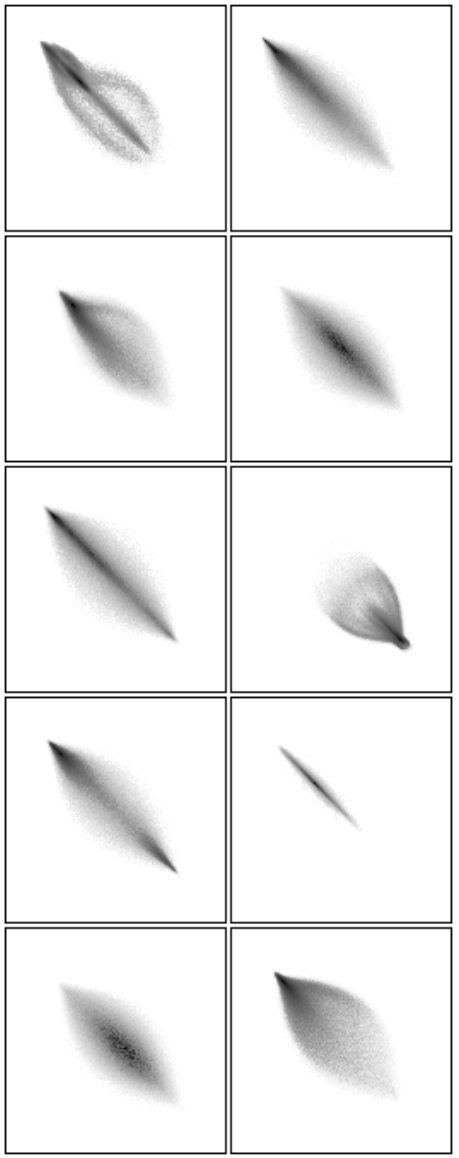

Figure B.1. Right-neighbor GLCM for Brodatz textures D1 to D10 (by row).

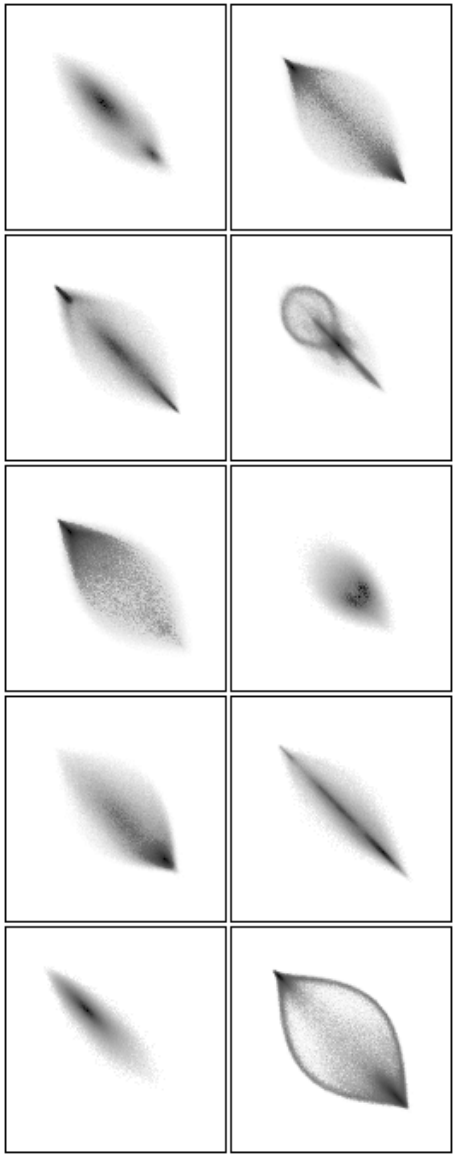

Figure B.2. Right-neighbor GLCM for Brodatz textures D11 to D20 (by row).

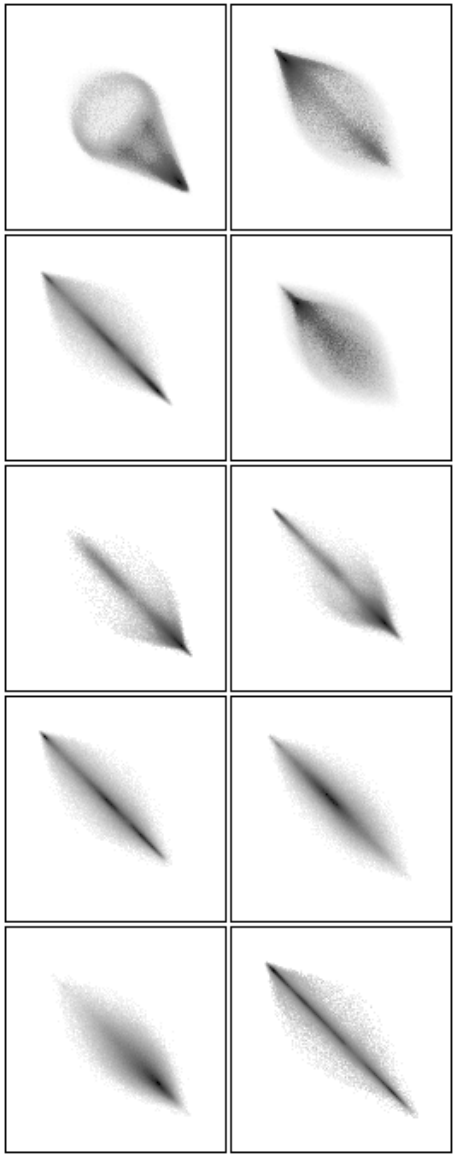

Figure B.3. Right-neighbor GLCM for Brodatz textures D21 to D30 (by row).

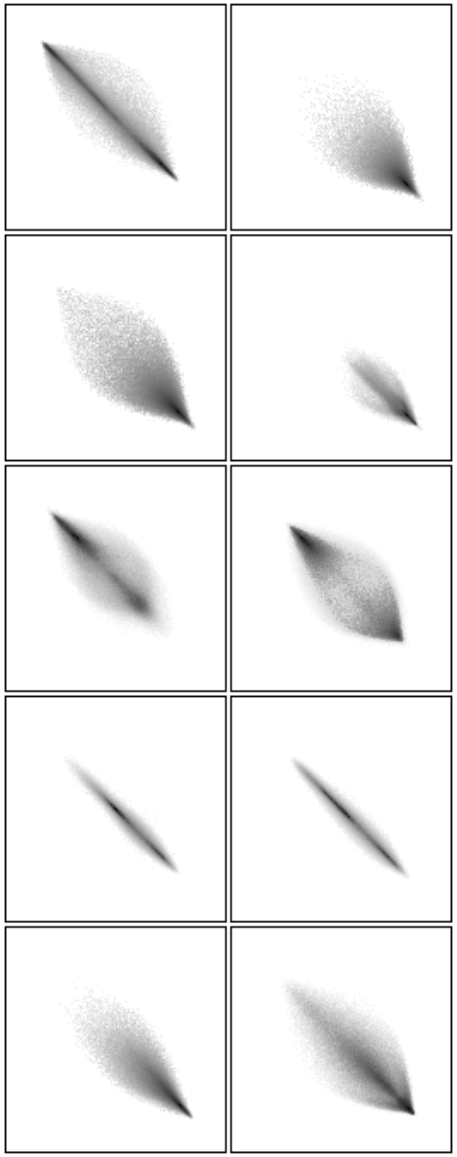

Figure B.4. Right-neighbor GLCM for Brodatz textures D31 to D40 (by row).

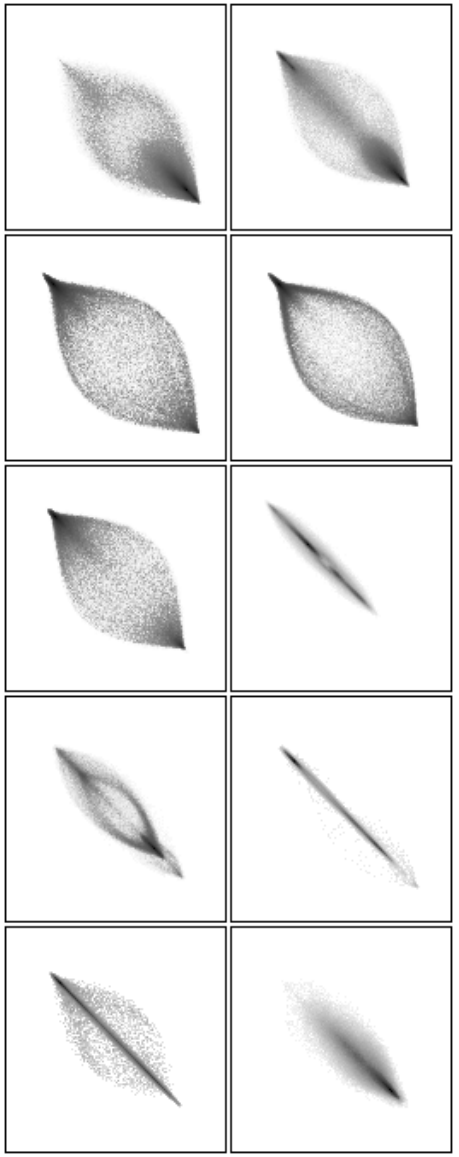

Figure B.5. Right-neighbor GLCM for Brodatz textures D41 to D50 (by row).

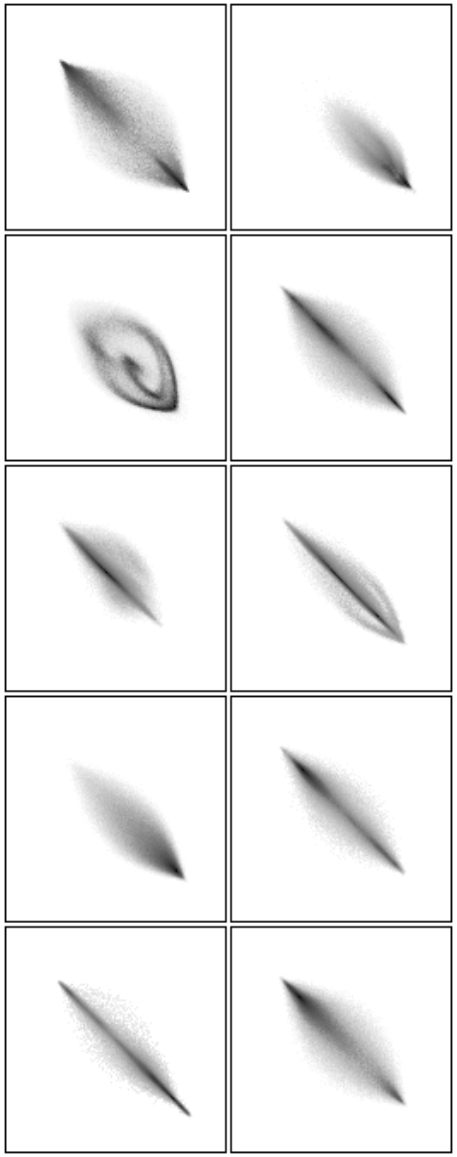

Figure B.6. Right-neighbor GLCM for Brodatz textures D51 to D60 (by row).

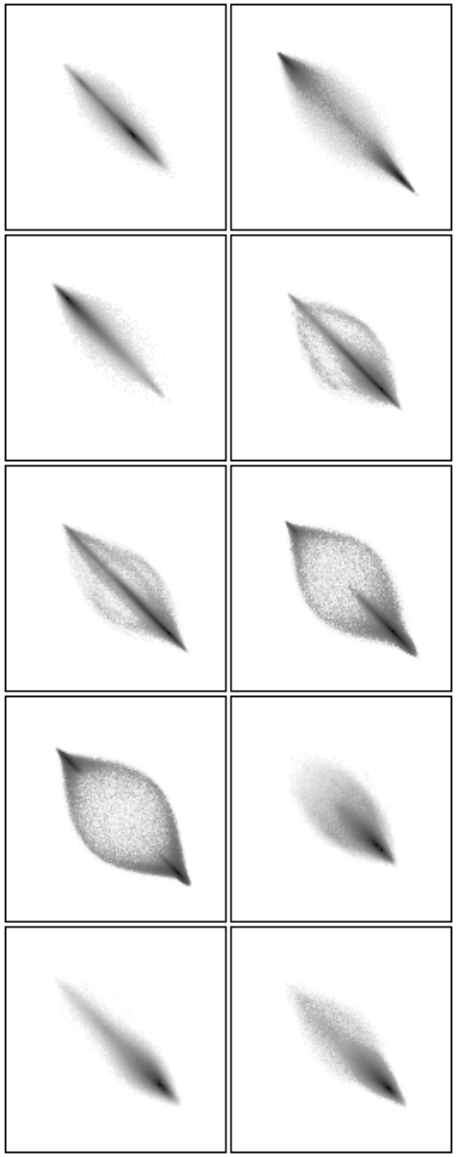

Figure B.7. Right-neighbor GLCM for Brodatz textures D61 to D70 (by row).

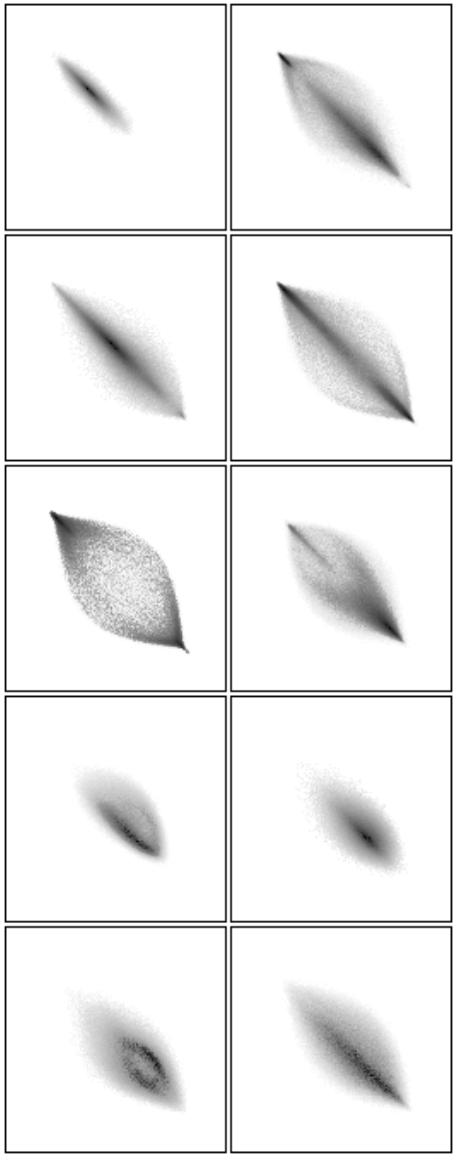

Figure B.8. Right-neighbor GLCM for Brodatz textures D71 to D80 (by row).

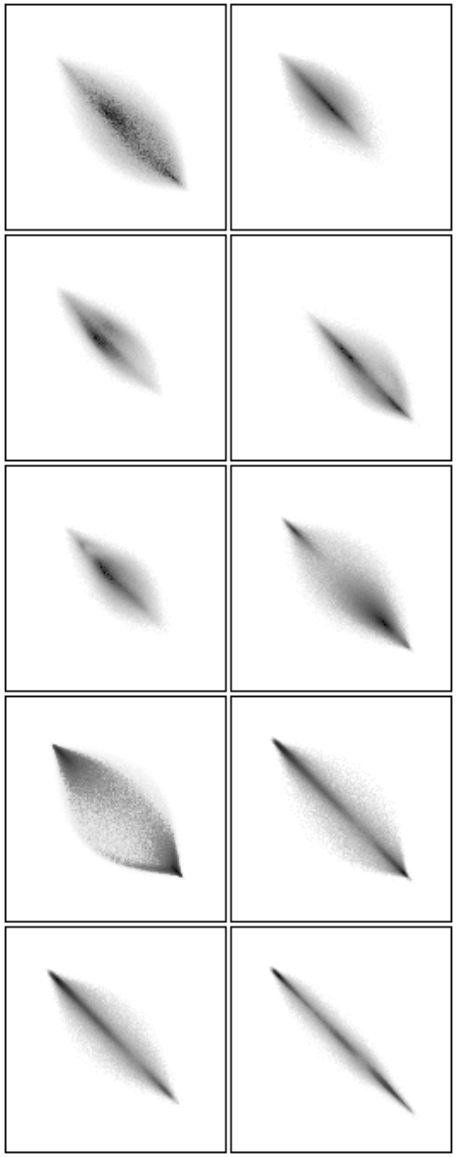

Figure B.9. Right-neighbor GLCM for Brodatz textures D81 to D90 (by row).

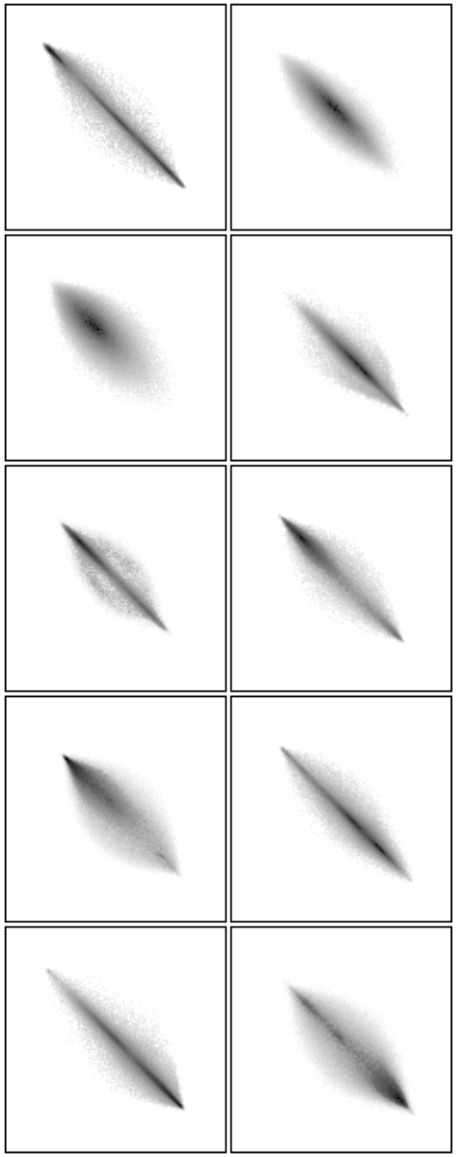

Figure B.10. Right-neighbor GLCM for Brodatz textures D91 to D100 (by row).

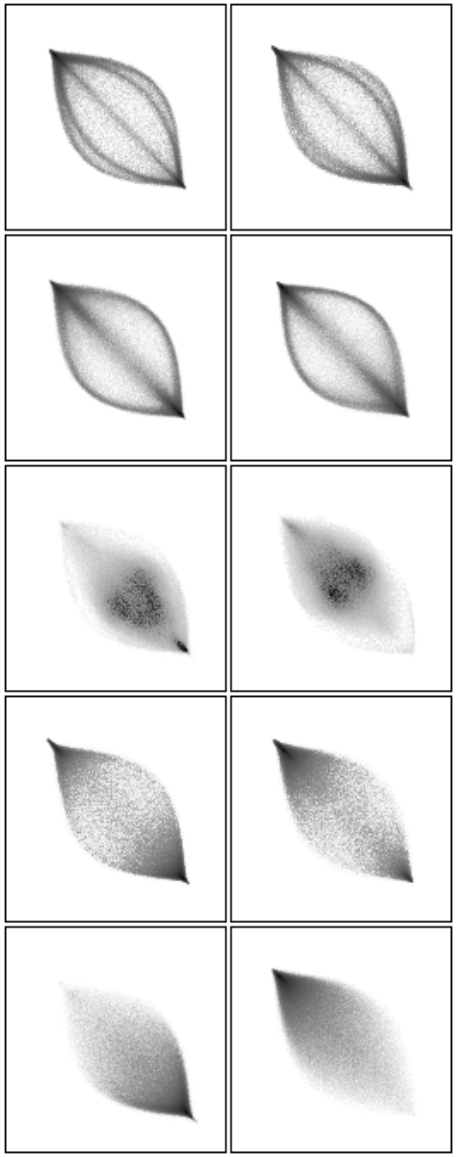

Figure B.11. Right-neighbor GLCM for Brodatz textures D101 to D110 (by row).

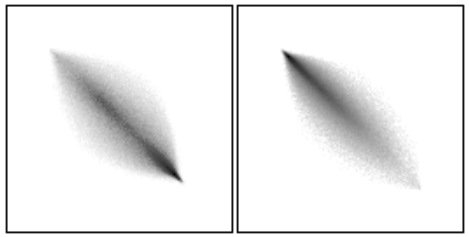

Figure B.12. Right-neighbor GLCM for Brodatz textures D111 and D112 (by row).

References

Aarts, E. & Korst, J. (1989). *Simulated Annealing and Boltzmann Machines.* J. Wiley & Sons, New York. 272 pp.

Adler, P. M., Jacquin, C. G., & Quiblier, J. A. (1990). Flow in simulated porous media. *Int. J. Multiphase Flow*, **16**(4), 691–712.

Åström, K. J. & Wittenmark, B. (1984). *Computer controlled systems: Theory and design.* Prentice-Hall International. 430 pp.

Berry, J. R. & Goutsias, J. (1991). A comparative study of matrix measures for maximum likelihood texture classification. *IEEE Transactions on Systems, Man, and Cybernetics*, **21**(1), 252–261.

Besag, J. & Moran, P. A. P. (1975). On the estimation and testing of spatial interaction in gaussian lattice processes. *Biometrika*, **62**(3), 555–562.

Besag, J. (1974). Spatial interaction and the statistical analysis of lattice systems. *Journal of the Royal Statistical Society, Series B*, **36**, 192–236.

Besag, J. (1975). Statistical analysis of non-lattice data. *The Statistician*, **24**, 179–195.

Besag, J. (1977). Efficiency of pseudolikelihood estimation for simple gaussian fields. *Biometrika*, **64**(3), 616–618.

Besag, J. (1986). On the statistical analysis of dirty pictures. *Journal of the Royal Statistical Society, Series B*, **48**(3), 259–302.

Besag, J. (1989). Towards Bayesian image analysis. *Journal of Applied Statistics*, **16**(3), 395–407.

Bishop, Y. M. M., Fienberg, S. E., & Holland, P. W. (1975). *Discrete Multivariate Analysis - Theory and Practice.* MIT Press, Cambridge, Massachusetts. 557 pp.

Breiman, L., Friedman, J. H., Olshen, R. A., & Stone, C. J. (1984). *Classification and regression trees.* Wadsworth & Brooks/Cole advanced books & software, Monterey, California. 358 pp.

Brodatz, P. (1966). *Textures - A Photographic Album for Artists and Designers.* Dover, New York.

Brush, S. G. (1967). History of the Lenz-Ising model. *Reviews of Modern Physics*, **39**(4), 883–893.

Burt, P. J. (1981). Fast filter transforms for image processing. *Computer Graphics and Image Processing*, **16**, 20–51.

Chellappa, R. (1985). Two-dimensional discrete gaussian Markov random field models for image processing. In Kanal, L. & Rosenfeld, A. (Eds.), *Progress in Pattern Recognition 2*, pp. 79–112. North-Holland, Amsterdam.

Cohen, J. (1960). A coefficient of agreement for nominal scales. *Educational and Psych. Meas.*, **20**, 37–46.

Conners, R. W. & Harlow, C. A. (1980). A theoretical comparison of texture algorithms. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **2**(3), 204–222.

Conners, R. W., Trivedi, M. M., & Harlow, C. A. (1984). Segmentation of a high-resolution urban scene using texture operators. *Computer Vision, Graphics and Image Processing*, **25**, 273–310.

Cross, G. R. & Jain, A. K. (1983). Markov random field texture models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **5**, 25–39.

Derin, H. & Elliot, H. (1987). Modelling and segmentation of noisy and textured images using Gibbs random fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **9**, 39–55.

du Buf, J. M. H., Kardan, M., & Spann, M. (1990). Texture feature performance for image segmentation. *Pattern Recognition*, **23**(3/4), 291–309.

Dubes, R. C. & Jain, A. K. (1989). Random field models in image analysis. *Journal of Applied Statistics*, **16**(2), 131–164.

Dubrule, O. (1989). A review of stochastic models for petroleum reservoirs. In Armstrong, M. (Ed.), *Geostatistics*, Vol. 2, pp. 493–506. Kluwer Academic Publishers.

Duda, R. O. & Hart, P. E. (1972). Use of the Hough transformation to detect lines and curves in pictures. *Communications of the acm*, **15**(1), 11–15.

Farmer, C. L. (1989). The mathematical generation of reservoir geology.. Paper presented at the Joint IMA/SPE European Conference on The Mathematics of Oil Recovery. Robinson College, Cambridge University, 25th-27th July.

Feller, W. (1968). *An introduction to probability theory and its applications*, Vol. 1. J. Wiley & Sons. 509 pp.

Figueiras-Vidal, A. R., Paez-Borrallo, J. M., & Garcia-Gomez, R. (1987). On using cooccurrence matrices to detect periodicities. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, **35**, 114–116.

Galloway, M. M. (1975). Texture analysis using gray level run lengths. *Computer Graphics and Image Processing*, **4**, 172–179.

Geman, S. & Geman, D. (1984). Stochastic relaxation, Gibbs distributions and the Bayesian restoration of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **6**, 721–741.

Geman, S. & Graffigne, C. (1987). Markov random fields and their applications to computer vision. In Gleason, A. (Ed.), *Proceedings of the International Congress of Mathematicians*, pp. 1496–1517 Berkeley, California.

Geman, S. & McClure, D. E. (1987). Statistical methods for tomographic image reconstruction. *Bulletin of the International Statistical Institute*, **52**, 5–21.

Geman, D., Geman, S., Graffigne, C., & Dong, P. (1990). Boundary detection by constrained optimization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **12**(7), 609–628.

Geman, D. (1990). Random fields and inverse problems in imaging. In *Saint-Flour lectures 1988*, Lecture Notes in Mathematics, pp. 113–193. Springer-Verlag.

Green, P. J. (1986). Contribution to the discussion of the paper by J. Besag. *Journal of the Royal Statistical Society, Series B*, **48**(3), 284–285.

Haldorsen, H. H., Brand, P. J., & Macdonald, C. J. (1988). Review of the stochastic nature of reservoirs. In Edwards, S. & King, P. (Eds.), *Mathematics in oil production*, No. 18 in The Institute of Mathematics & its Applications conference series, pp. 109–209. Clarendon Press, Oxford.

Haralick, R. M., Shanmugam, K., & Dinstein, I. (1973). Textural features for image classification. *IEEE Transactions on Systems, Man, and Cybernetics*, **3**(6), 610–621.

Haralick, R. M., Sternberg, S. R., & Zhuang, X. (1987). Image analysis using mathematical morphology. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **9**(4), 532–550.

Haralick, R. M. (1979). Statistical and structural approaches to texture. *Proceedings of the IEEE*, **67**(5), 786–804.

Hassner, M. & Sklansky, J. (1980). The use of Markov random fields as models of texture. *Computer Graphics and Image Processing*, **12**, 357–370.

Ising, E. (1925). Beitrag zür Theorie des Ferromagnetismus. *Zeitschrift für Physik*, **31**, 253–258.

Jaynes, E. T. (1957). Information theory and statistical mechanics. *Physical Review*, **106**, 620–630.

Julesz, B. & Bergen, J. R. (1983). Textons, the fundamental elements in preattentive vision and perception of textures. *Bell Syst. Tech. J.*, **62**(6), 1619–1645.

Julesz, B. (1975). Experiments in the visual perception of texture. *Scientific American*, **232**(4), 34–43.

Julesz, B. (1981). Textons, the elements of texture perception, and their interactions. *Nature*, **290**, 91–97.

Kashyap, R. L. & Chellappa, R. (1983). Estimation and choice of neighbors in spatial interaction models of images. *IEEE Transactions on Information Theory*, **29**(1), 60–72.

Kashyap, R. L., Chellappa, R., & Khotanzad, A. (1982). Texture classification using features derived from random field models. *Pattern Recognition Letters*, **1**(1), 43–50.

Kinderman, R. & Snell, J. L. (1980). *Markov Random Fields and their Applications*. American Mathematical Society, Providence, Rhode Island. 142 pp.

Kirkland, M. (1989). *Simulation methods for Markov random fields*. Ph.D. thesis, University of Strathclyde, Glasgow. 189 pp.

Knuth, D. E. (1973). *The Art of Computer Programming*, Vol. 2: Seminumerical Algorithms. Addison-Wesley.

Laws, K. I. (1980). *Textured Image Segmentation*. USCIPI report # 940, Signal and Image Processing Institute, University of Southern California, Los Angeles. 178 pp.

Linfoot, E. H. (1957). An informational measure of correlation. *Information and Control*, **1**, 85–89.

Liu, S. & Jernigan, M. E. (1990). Texture analysis and discrimination in additive noise. *Computer Vision, Graphics and Image Processing*, **49**, 52–67.

Margalit, A. (1989). A parallel algorithm to generate to generate a Markov random field image on a SIMD hypercube machine. *Pattern Recognition Letters*, **9**(4), 263–278.

Marroquin, J., Miter, S., & Poggio, T. (1987). Probabilistic solution of ill-posed problems in computational vision. *Journal of the American Statistical Association*, **82**, 76–89.

Metropolis, N., Rosenbluth, A. W., Rosenbluth, M. N., Teller, A. H., & Teller, E. (1953). Equation of state calculations by fast computing machines. *J. Chem. Phys.*, **21**, 1087–1092.

Murray, D. W., Kashko, A., & Buxton, H. (1986). A parallel approach to the picture restoration algorithm of Geman and Geman on an SIMD machine. *Image and Vision Computing*, **4**(3), 133–142.

Onsager, L. (1944). Crystal statistics. I. a two-dimensional model with an order-disorder transition. *Physical Review*, **65**, 117–149.

Parkkinen, J. & Oja, E. (1986). Cooccurrence matrices and subspace methods in texture analysis. In *Proc. 8th ICPR*, pp. 405–408 Paris, France.

Parkkinen, J., Selkäinaho, K., & Oja, E. (1990). Detecting texture periodicity from the cooccurrence matrix. *Pattern Recognition Letters*, **11**(1), 43–50.

Pickard, D. K. (1987). Inference for discrete Markov fields: the simplest nontrivial case. *Journal of the American Statistical Association*, **82**(397), 90–96.

Potts, R. B. (1952). Some generalized order-disorder transformations. *Proc. Camb. Phil.*, **48**, 106–109.

Press, W. H., Flannery, B. P., Teukolsky, S. A., & Vetterling, W. T. (1988). *Numerical Recipes in C*. Cambridge University Press. 735 pp.

Rao, A. R. (1990). *A Taxonomy for Texture Description and Identification*. Springer-Verlag. 197 pp.

Ripley, B. D. & Kirkland, M. D. (1990). Iterative simulation methods. *J. Comp. Appl. Math.*, **31**, 165–172.

Ripley, B. D. (1981). *Spatial Statistics*. J. Wiley & Sons. 252 pp.

Ripley, B. D. (1987). *Stochastic simulation*. J. Wiley & Sons. 237 pp.

Ripley, B. D. (1988). *Statistical inference for spatial processes*. Cambridge University Press. 148 pp.

Ripley, B. D. (1992). Stochastic models for the distribution of rock types in petroleum reservoirs. In Walden, A. & Guttorp, P. (Eds.), *Statistics in the Environmental and Earth Sciences*. Griffin.

Seneta, E. (1981). *Non-negative Matrices and Markov Chains*. Springer-Verlag, New York. 279 pp.

Serra, J. (1982). *Image Analysis and Mathematical Morphology*, Vol. 1. Academic Press, New York. 610 pp.

Serra, J. (Ed.). (1988). *Image Analysis and Mathematical Morphology*, Vol. 2. Academic Press, New York. 411 pp.

Siew, L. H., Hodgson, R. M., & Wood, E. J. (1988). Texture measures for carpet wear assessment. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **10**(1), 92–105.

Sternberg, S. R. (1986). Grayscale morphology. *Computer Vision, Graphics and Image Processing*, **35**, 333–355.

Sun, C. & Wee, W. G. (1983). Neighboring gray level dependence matrix for texture classification. *Computer Vision, Graphics and Image Processing*, **23**, 341–352.

Swendsen, R. H. & Wang, J. (1987). Nonuniversal critical dynamics in Monte Carlo simulations. *Physical Review Letters*, **58**(2), 86–88.

Tomita, F. & Tsuji, S. (1990). *Computer Analysis of Visual Texture*. Kluwer Academic Publishers. 173 pp.

Unser, M. (1986a). Local linear transforms for texture measurements. *Signal Processing*, **11**, 61–79.

Unser, M. (1986b). Sum and difference histograms for texture classification. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **8**(1), 118–125.

van Gool, L., Dewaele, P., & Oosterlinck, A. (1985). Texture analysis anno 1983. *Computer Vision, Graphics and Image Processing*, **29**, 336–357.

Vickers, A. L. & Modestino, J. W. (1982). A maximum likelihood approach to texture classification. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **4**(1), 61–68.

Weszka, J. S., Dyer, C. R., & Rosenfeld, A. (1976). A comparative study of texture measures for terrain classification. *IEEE Transactions on Systems, Man, and Cybernetics*, **6**(4), 269–285.

Wolff, U. (1989). Collective Monte Carlo updating for spin systems. *Physical Review Letters*, **62**, 361–364.

Wu, F. Y. (1982). The Potts model. *Reviews of Modern Physics*, **54**(1), 235–268.

Yang, C. N. (1952). The spontaneous magnetization of the two-dimensional ising model. *Physical Review*, **85**, 808–816.

Zucker, S. W. & Terzopoulos, D. (1980). Finding structure in co-occurrence matrices for texture analysis. *Computer Graphics and Image Processing*, **12**, 286–308.

Index